

EXHIBIT G

PacBioFileFormats Documentation

Release 11.0.0

Pacific Biosciences

Feb 02, 2022

Contents

1	Brief primer and lexicon for PacBio SMRT sequencing	3
2	BAM format specification for PacBio	5
3	PacBio BAM index file (<code>bam.pbi</code>) format	17
4	FASTA file format	23
5	DataSet format specification	25
6	Run Design CSV specification for PacBio	53
7	Legacy formats	59
8	APIs available	75
9	Data Model XSD	77

As of the 3.0 release of SMRTanalysis, PacBio is embracing the industry standard BAM format for (both aligned and unaligned) basecall data files. We have also formulated a BAM companion file format (*bam.pbi*) enabling fast access to a richer set of per-read information as well as compatibility for software built around the legacy *cmp.h5* format.

CHAPTER 1

Brief primer and lexicon for PacBio SMRT sequencing

PacBio SMRT sequencing operates within a silicon chip (a **SMRTcell**) fabricated to contain a large number of microscopic holes (**ZMWs**, or **zero-mode waveguides**), each assigned a **hole number**.

Within a ZMW, PacBio SMRT sequencing is performed on a circularized molecule called a **SMRTbell**. The SMRTbell, depicted below, consists of:

- the customer's double-stranded DNA **insert** (with sequence I , read following the arrow)
- (optional) double-stranded DNA **barcodes** (sequences B_L, B_R) used for multiplexing DNA samples. While the barcodes are optional, they must be present at both ends if present at all. Barcodes may or may not be *symmetric*, where symmetric means $B_L = B_R^{RC}$.
- SMRTbell **adapters** (sequences A_L, A_R), each consisting of a double stranded stem and a single-stranded hair-pin loop. Adapters may or may not be *symmetric*, where symmetric means $A_L = A_R$.

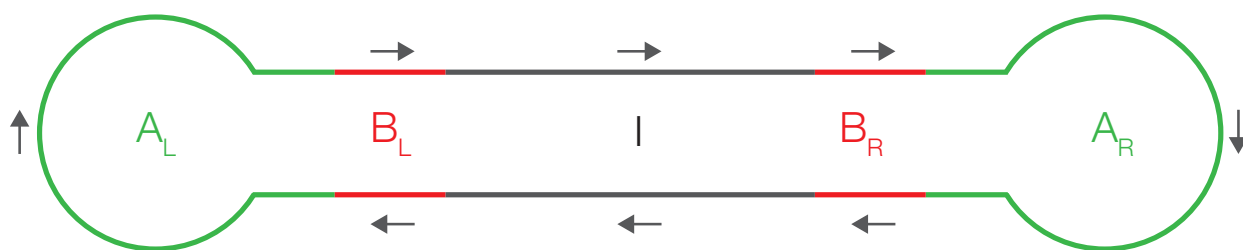


Fig. 1: A schematic drawing of a SMRTbell

SMRT sequencing interrogates the incorporated bases in the product strand of a replication reaction. Assuming the sequencing of the template above began at START, the following sequence of bases would be incorporated (where we are using the superscripts C, R, and RC to denote sequence complementation, reversal, and reverse-complementation):

$$A_L^C B_L^C I^C B_R^C A_R^C B_R^R I^R B_L^R A_L^C \dots$$

(note the identity $(x^{RC})^C = x^R$).

The **ZMW read** is the full output of the instrument/basecaller upon observing this series of incorporations, subject to errors due to optical and other limitations. **Adapter regions** and **barcode regions** are the spans of the ZMW read

corresponding to the adapter and barcode DNA. The **subreads** are the spans of the ZMW read corresponding to the DNA insert.

One complication arises when one considers the possibility that a ZMW might not contain a single sequencing reaction. Indeed it could contain zero—in which case the ensuing basecalls are a product of background noise—or it could contain more than one, in which case the basecall sequence represents two intercalated reads, effectively appearing as noise. To remove such noisy sequence, the **high quality (HQ) region finder** in PostPrimary algorithmically detects a maximal interval of the ZMW read where it appears that a single sequencing reaction is taking place. This region is designated the **HQ region**, and in the standard mode of operation, PostPrimary will only output the subreads detected within the HQ region.

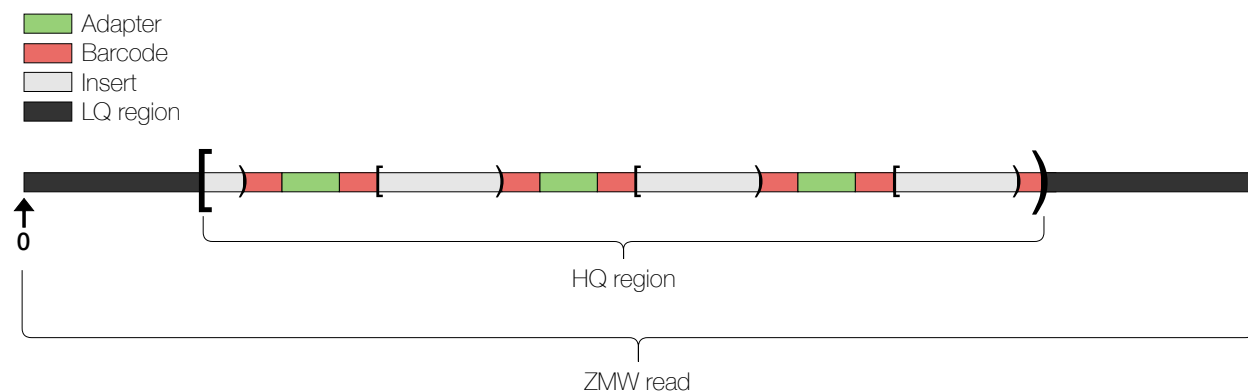


Fig. 2: A schematic of the regions designated within a ZMW read

Note: Our coordinate system begins at the first basecall in the ZMW read (deemed base 0)—i.e., it is *not* relative to the HQ region. Intervals in PacBio reads are given in end-exclusive (“half-open”) coordinates. This style of coordinate system should be familiar to Python or C++ STL programmers.

1.1 BAM everywhere

Unaligned BAM files representing the *subreads* will be produced natively by the PacBio instrument. The subreads BAM will be the starting point for secondary analysis. In addition, the artifacts resulting from removing adapter and LQ region sequences will be retained in a `scraps.bam` file.

The circular consensus tool/workflow (CCS) will take as input an unaligned subreads BAM file and produce an output BAM file containing unaligned *consensus* reads.

Alignment (mapping) programs take these unaligned BAM files as input and will produce *aligned* BAM files, faithfully retaining all tags and headers.

CHAPTER 2

BAM format specification for PacBio

The BAM format is a binary, compressed, record-oriented container format for raw or aligned sequence reads. The associated SAM format is a text representation of the same data. The [specifications for BAM/SAM](#) are maintained by the SAM/BAM Format Specification Working Group.

PacBio-produced BAM files are fully compatible with the BAM specification. In this document we describe the way we make use of the extensibility mechanisms of the BAM specification to encode PacBio-specific information, as well as conventions we adhere to.

An example file adhering to this specification will be maintained in the *pbcore* Python library.

2.1 Version

The PacBio BAM specification version described here is 5.0.0. PacBio BAM files adhering to this spec contain the tag `pb:5.0.0` in the `@HD` header.

2.2 Coordinate conventions

The BAM format uses a 0-based coordinate system to refer to positions and intervals on the reference.

PacBio also uses a 0-based coordinate system to refer to positions and intervals within sequence reads. Positions in PacBio reads are reckoned from the first ZMW read base (as base 0), *not* the first base in the HQ region.

Perhaps confusingly, the text SAM format uses 1-based coordinate system.

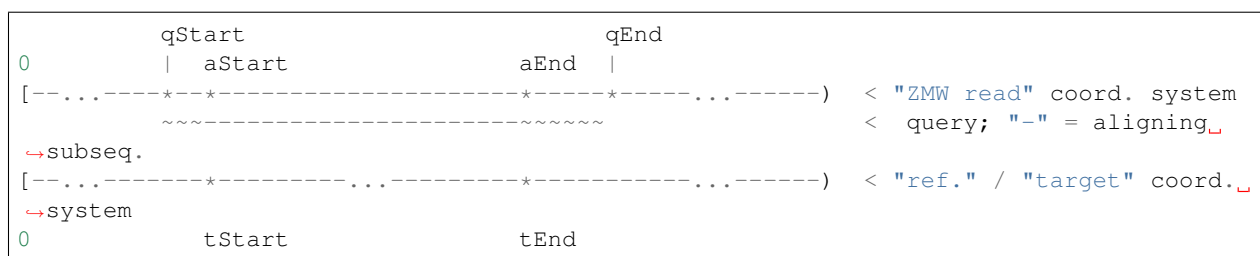
Note that following the SAM/BAM specification, 0-based coordinate intervals are defined as half-open (end exclusive) while 1-based intervals are closed.

2.3 Query versus *aligned query* terminology

A sequence read presented to an aligner is termed a *query*; typically this query will be a subsequence of an entire PacBio ZMW read—most commonly, it will be a *subread*, which is basecalls from a single pass of the insert DNA molecule.

Upon alignment, generally only a subsequence of the query will align to the reference genome, and that subsequence is referred to as the *aligned query*. Under *soft-clipping*, the entirety of the query is stored in the aligned BAM, but the CIGAR field indicates that some bases at either end are excluded from the alignment.

Abstractly, we denote the extent of the *query* in ZMW read as $[qStart, qEnd)$ and the extent of the aligned subinterval as $[aStart, aEnd)$. The following graphic illustrates these intervals:



In our BAM files, the $qStart$, $qEnd$ are contained in the qs and qe tags, (and reflected in the $QNAME$); the bounds of the *aligned query* in the ZMW read can be determined by adjusting qs and qe by the number of soft-clipped bases at the ends of the alignment (as found in the CIGAR).

Note: In the legacy cmp.h5 file format, soft-clipping was not possible, and the bounds of the original query were not stored. Only $aStart$, $aEnd$ were stored, although in that file format they were referred to as $rStart$, $rEnd$.

2.4 QNAME convention

By convention the $QNAME$ (“query template name”) for unrolled reads and subreads is in the following format:

```
{movieName}/{holeNumber}/{qStart}_{qEnd}
```

where $[qStart, qEnd)$ is the 0-based coordinate interval representing the span of the *query* in the ZMW read, as above.

For CCS reads, the $QNAME$ convention is:

```
{movieName}/{holeNumber}/ccs
```

In “by-strand mode” the $QNAME$ for CCS reads may include a suffix *fwd* or *rev* to convey strand information:

```
{movieName}/{holeNumber}/ccs/fwd
{movieName}/{holeNumber}/ccs/rev
```

2.5 CIGAR conventions

The “M” CIGAR op (BAM_CMATCH) is *forbidden* in PacBio BAM files. PacBio BAM files use the more explicit ops “X” (BAM_CDIFF) and “=” (BAM_CEQUAL). PacBio software will abort if BAM_CMATCH is found in a CIGAR field.

2.6 BAM filename conventions

Since we will be using BAM format for different kinds of data, we will use a `suffix.bam` filename convention:

Data type	Filename template
ZMW reads from movie	<i>movieName.zmws.bam</i>
Analysis-ready subreads ¹ from movie	<i>movieName.subreads.bam</i>
Excised adapters, barcodes, and rejected sub-reads	<i>movieName.scrap.s.bam</i>
CCS reads computed from movie	<i>movieName.ccs.bam</i>
Aligned subreads in a job	<i>jobID.aligned_subreads.bam</i>
Aligned CCS in a job	<i>jobID.aligned_ccs.bam</i>

¹ Data in a `subreads.bam` file should be `analysis ready`, meaning that all of the data present is expected to be useful for down-stream analyses. Any subreads for which we have strong evidence will not be useful (e.g. double-adapter inserts, single-molecule artifacts) should be excluded from this file and placed in `scrap.s.bam` as a `Filtered` with an `SC` tag of `F`.

2.7 BAM sorting conventions

Aligned PacBio BAM files shall be sorted by position in the standard fashion as done by `samtools sort`. The `BAM @HD : : SO` tag shall be set to `coordinate`.

Unaligned PacBio BAM files shall be sorted by `QNAME`, so that all subreads from a ZMW hole are stored contiguously in a file, with groups by ZMW hole number in numerical order, and within a ZMW, numerically by `qStart`. In case subreads and CCS reads are combined in a BAM, the CCS reads will sort after the subreads (`ccs` follows `{qStart}_{qEnd}`). Note that this sorting is not strictly alphabetical, so we shall set the `BAM @HD : : SO` tag to `unknown`.

2.8 Use of headers for file-level information

Beyond the usual information encoded in headers that is called for SAM/BAM spec, we encode special information as follows.

@RG (read group) header entries:

ID tag (identifier): contains an 8-character string interpretable as the hexadecimal representation of an integer. Optionally, a read group identifier may contain barcode labels to distinguish demultiplexed samples. Read groups should have distinct `ID` values.

Note: Standard read group identifiers for PacBio data are calculated as follows:

```
RGID_STRING := md5(movieName + "/" + readType)[:8]
```

where *movieName* is the moviename (@RG::PU) and *readType* is the read type (found in @RG::DS). Note that *movieName* is lowercase while *readType* is uppercase. *md5* is understood to be the (lowercase) hex md5 digest of the input string.

Optionally for *readType* CCS, strandness can be encoded in the ID. This is to ensure that multiple types of reads, double- and single- stranded, can be stored in the same BAM file, without hole number collisions in the PacBio BAM index file. The RGID_STRING is then defined as:

```
RGID_STRING := md5(movieName + "/" + readType + "/" + strand)[:8]
```

where strand must be lowercase fwd or rev; it may not be empty.

The RGID_INT is defined as:

```
RGID_INT := int32.Parse(RGID_STRING)
```

RGID_STRING is used in the @RG header and in the RG tag of BAM records, while RGID_INT is used in the PacBio BAM index file.

Note that RGID_INT may be negative.

Example: CCS reads for a movie named “movie32” would have

- RGID_STRING = “f5b4ffb6”
- RGID_INT = -172687434

Optional barcode labels must be appended to the RGID_STRING as follows:

```
{RGID_STRING}/{bcForward}--{bcReverse}
```

where the bcForward and bcReverse labels correspond to the 0-based positions in the FASTA file of barcodes. These are the same values used to populate a barcoded record’s bc tag.

PL tag (“platform”): contains "PACBIO".

PM tag (“platform model”): contains "ASTRO", "RS", or "SEQUEL", reflecting the PacBio instrument series.

PU tag (“platform unit”): contains the PacBio movie name.

LB tag (“Well Sample Name”): contains the user-supplied name of the library.

SM tag (“Bio Sample Name”): contains the user-supplied name of the biological sample.

BC tag (“barcodes”): contains the barcode sequences associated with this read group. This tag is not required in all PacBio BAM files, but must be provided when the read group ID includes barcode labels.

The value must be represented in the format recommended by the SAM/BAM spec. Barcode *sequences* will be concatenated by a single dash. If both barcodes are the same, only one needs to be provided.

```
{seq} {seq1}-{seq2}
```

Note that this differs from the format used to label barcode indices on a read group’s ID.

DS tag (“description”): contains some semantic information about the reads in the group, encoded as a semicolon-delimited list of “Key=Value” strings, as follows:

Mandatory items:

Key	Value spec	Value example
READTYPE	One of ZMW, HQREGION, SUB-READ, CCS, SCRAP, or UNKNOWN	SUBREAD
BINDINGKIT	Binding kit part number	100236500
SEQUENCINGKIT	Sequencing kit part number	001558034
BASECALLERVERSION	Basecaller version number	2.1
FRAMERATEHZ	Frame rate in Hz	100
CONTROL	TRUE if reads are classified as spike-in controls, otherwise CONTROL key is absent	TRUE
STRAND	Stores strandness of single-stranded reads as FORWARD or REVERSE. Key is absent if reads are double-stranded. Only applies to READTYPE CCS.	FORWARD

Note: The READTYPE values encountered in secondary analysis will be limited to SUBREAD and CCS. The remaining READTYPE values will only be encountered in intermediate steps before secondary analysis.

Base feature manifest—absent item means feature absent from reads:

Key	Value spec	Value example
DeletionQV	Name of tag used for DeletionQV	dq
DeletionTag	Name of tag used for DeletionTag	dt
InsertionQV	Name of tag used for InsertionQV	iq
MergeQV	Name of tag used for MergeQV	mq
SubstitutionQV	Name of tag used for SubstitutionQV	sq
SubstitutionTag	Name of tag used for SubstitutionTag	st
Ipd:Frames	Name of tag used for IPD, in raw frame count.	ip
Ipd:CodecV1	Name of tag used for IPD, compressed according to Codec V1.	ip
PulseWidth:Frames	Name of tag used for PulseWidth, in raw frame count.	pw
PulseWidth:CodecV1	Name of tag used for PulseWidth, compressed according to Codec V1.	pw

Optional items:

Note: These items are optional if there are no “bc” tags in the reads belonging to this read-group, otherwise they are mandatory.

Key	Value spec	Value example
Bar-codeFile	Name of the Fasta file containing the sequences of the barcodes used	pacbio_384_barcodes.fasta
Bar-code-Hash	The MD5 hash of the contents of the barcoding sequence file, as generated by the <i>md5sum</i> commandline tool	0a294bb959fc6c766967fc8beeb4d88d
Bar-code-Count	The number of barcode sequences in the Barcode File	384
Bar-code-Mode	Experimental design of the barcodes Must be Symmetric/Asymmetric/Tailed or None	Symmetric
BarcodeQuality	The type of value encoded by the bq tag Must be Score/Probability/None	Probability

2.9 Use of read tags for per-read information

Tag	Type	Description
qs	i	0-based start of query in the ZMW read (absent in CCS)
qe	i	0-based end of query in the ZMW read (absent in CCS)
ws	i	Start of first base of the query ('qs') in approximate raw frame count since start of movie. For a CCS read, the start of the first base of the first incorporated subread.
we	i	Start of last base of the query ('qe - 1') in approximate raw frame count since start of movie. For a CCS read, the start of the last base of the last incorporated subread.
zm	i	ZMW hole number
np	i	NumPasses (1 for subreads, variable for CCS—encodes number of <i>complete</i> passes of the insert)
ec	f	Effective coverage for CCS reads, the average subread coverage across all windows (only present in CCS reads)
rq	f	Float in [0, 1] encoding expected accuracy
sn	B,f	4 floats for the average signal-to-noise ratio of A, C, G, and T (in that order) over the HQRegion

2.10 Use of read tags for per-read-base information

The following read tags encode features measured/calculated per-basecall. Unlike *SEQ* and *QUAL*, aligners will not orient these tags. They will be maintained in *native* orientation (in the same order and sense as collected from the instrument) even if the read record has been aligned to the reverse strand.

Tag	Type	Description
dq	Z	DeletionQV
dt	Z	DeletionTag
iq	Z	InsertionQV
mq	Z	MergeQV
sq	Z	SubstitutionQV
st	Z	SubstitutionTag
ip	B,C or B,S	IPD (raw frames or codec V1)
pw	B,C or B,S	PulseWidth (raw frames or codec V1)

Notes:

- QV metrics are ASCII+33 encoded as strings
- *DeletionTag* and *SubstitutionTag* represent alternate basecalls, or “N” when there is no alternate basecall available. In other words, they are strings over the alphabet “ACGTN”.
- The IPD (interpulse duration) value associated with a base is the number of frames *preceding* its incorporation, while the PW (pulse width) is the number of frames during its incorporation.
- Encoding of kinetics features (*ip*, *pw*) is described below.

2.11 Use of read tags for HiFi per-read-base kinetic information

The following read tags contain averaged kinetic information (IPD/PulseWidth) from subreads when applying CCS to generate HiFi reads. These are computed and stored independently for both orientations of the insert. Forward is defined & stored with respect to the orientation represented in *SEQ* and is considered to be the native orientation. Reverse tags are stored in the opposite direction, e.g. from the last base to the first. As with other PacBio-specific tags, aligners will not re-orient these fields.

Tag	Type	Description
fi	B,C	Forward IPD (codec V1)
ri	B,C	Reverse IPD (codec V1)
fp	B,C	Forward PulseWidth (codec V1)
rp	B,C	Reverse PulseWidth (codec V1)
fn	i	Forward number of complete passes (zero or more)
rn	i	Reverse number of complete passes (zero or more)

The following clipping example illustrates the coordinate system for these tags, shown as stored in the BAM file:

Original

```

SEQ:  A  A  C  C  G  T  T  A  G  C
fi/fp: f0, f1, f2, f3, f4, f5, f6, f7, f8, f9
ri/rp: r9, r8, r7, r6, r5, r4, r3, r2, r1, r0

```

Clipped to [1, 4)

```

SEQ:  A  C  C
fi/fp: f1, f2, f3
ri/rp: r3, r2, r1

```

Notes:

- When CCS filtering is disabled, no averaging occurs with ZMWs that don’t have enough passes to generate HiFi reads. Instead, the *pw/ip* values are passed as is from a representative subread.
- Minor cases exist where a certain orientation may get filtered out entirely from a ZMW, preventing valid values from being passed for that record. In these cases, empty lists will be passed for the respective record/orientation and number of passes will be set to zero.

- Flanking zeroes in kinetics arrays should be ignored for the respective strand. For instance, when SEQ is AAACGCGTTT and fp:B:C,0,0,0,3,4,5,6,0,0,0, then any downstream application should only use CGCG in its analysis, and ignore the AAA and TTT stretches.

2.12 Use of read tags for per-read-base base modifications

The following read tags encode base modification information. Base modifications are encoded according to the [SAM tags specifications](#) and any conflict is unintentional.

Tag	Type	Description
Mm	Z	Base modifications / methylation
Ml	B,C	Base modification probabilities

Notes:

- For informational purposes only: The continuous probability range of 0.0 to 1.0 is remapped to the discrete integers 0 to 255 inclusively in the Ml tag. The probability range corresponding to an integer N is $N/256$ to $(N + 1)/256$.

2.13 How to annotate scrap reads

Reads that belong to a read group with READTYPE=SCRAP have to be annotated in a hierarchical fashion:

- 1) Classification with tag sz occurs on a per ZMW level, distinguishing between spike-in controls, sentinels of the basecaller, malformed ZMWs, and user-defined templates.
- 2) A region-wise annotation with tag sc to label adapters, barcodes, low-quality regions, and filtered subreads.

Tag	Type	Description
sz	A	ZMW classification annotation, one of N:=Normal, C:=Control, M:=Malformed, or S:=Sentinel ¹
sc	A	Scrap region-type annotation, one of A:=Adapter, B:=Barcode, L:=LQRegion, or F:=Filtered ²

¹ reads in the subreads/hqregions/zmws.bam file are implicitly marked as Normal, as they stem from user-defined templates.

² sc tags 'A', 'B', and 'L' denote specific classes of non-subread data, whereas the 'F' tag is reserved for subreads that are undesirable for downstream analysis, e.g., being artifactual or too short.

2.14 QUAL

The QUAL field in BAM alignments is intended to reflect the reliability of a basecall, using the Phred-encoding convention, as described in the [SAM spec](#).

Both CCS and raw read BAM files respect this convention; historically, and for the present moment, the encoded probability reflects the confidence of a basecall against alternatives including substitution, deletion, and insertion.

We expect that more details will follow here in a later spec revision.

2.15 Subread local context

Some algorithms can make use of knowledge that a subread was flanked on both sides by adapter or barcode hits, or that the subread was in one orientation or the other (as can be deduced when asymmetric adapters or barcodes are used).

To facilitate such algorithms, we furnish the `cx` bitmask tag for subread records. The `cx` value is calculated by binary OR-ing together values from this flags enum:

```
enum LocalContextFlags
{
    ADAPTER_BEFORE      = 1,
    ADAPTER_AFTER       = 2,
    BARCODE_BEFORE      = 4,
    BARCODE_AFTER       = 8,
    FORWARD_PASS        = 16,
    REVERSE_PASS        = 32,
    ADAPTER_BEFORE_BAD  = 64,
    ADAPTER_AFTER_BAD   = 128
};
```

Orientation of a subread (designated by one of the mutually exclusive `FORWARD_PASS` or `REVERSE_PASS` bits) can be reckoned only if either the adapters or barcode design is asymmetric, otherwise these flags must be left unset. The convention for what is considered a “forward” or “reverse” pass is determined by a per-ZMW convention, defining one element of the asymmetric barcode/adaptor pair as the “front” and the other as the “back”. It is up to tools producing the BAM to determine whether to use adapters or barcodes to reckon the orientation, but if pass directions cannot be confidently and consistently assessed for the subreads from a ZMW, neither orientation flag should be set. Tools consuming the BAM should be aware that orientation information may be unavailable for subreads in a ZMW, but if is available for any subread in the ZMW, it will be available for all subreads in the ZMW.

The `ADAPTER_*` and `BARCODE_*` flags reflect whether the subread is flanked by adapters or barcodes at the ends.

The `ADAPTER_BEFORE_BAD` and `ADAPTER_AFTER_BAD` flags indicate that one or both adapters flanking this subread do not align to the adapter reference sequence(s). The adapter on this flank could be missing from the pbell molecule, or obscured by a local decrease in accuracy. Likewise, some nearby barcode or insert bases may be missing or obscured. `ADAPTER_*_BAD` flags can not be set unless the corresponding `ADAPTER_*` flag is set.

This tag is mandatory for subread records, but will be absent from non-subread records (scraps, ZMW read, CCS read, etc.)

Tag	Type	Description
<code>cx</code>	i	Subread local context Flags

2.16 Missing adapter annotation in CCS reads

The `ma` and `ac` tags indicate whether the molecule that produces a CCS read is missing a SMRTbell adapter on its left/start or right/end. The tags are produced by CCS version 6.3.0 and newer based on the `ADAPTER_BEFORE_BAD` and `ADAPTER_AFTER_BAD` information in the subread `cx` tag.

Tag	Type	Description
ac	B,i	Array containing four counts, in order: - detected adapters on left/start - missing adapters on left/start - detected adapters on right/end - missing adapter on right/end
ma	i	Bitmask storing if an adapter is missing on either side of the molecule. A value of 0 indicates neither end has a confirmed missing adapter. - 0x1 if adapter is missing on left/start - 0x2 if adapter is missing on right/end

2.17 Barcode analysis

In multiplexed workflows, we record per-subread tags representing the barcode call and a score representing the confidence of that call. The actual data used to inform the barcode calls—the barcode sequences and associated pulse features—will be retained in the associated `scraps.bam` file.

Tag	Type	Description
bc	B,S	Barcode Calls (per-ZMW)
bq	i	Barcode Quality (per-ZMW)

- Both the `bc` and `bq` tags are calculated per-ZMW, so every subread belonging to a given ZMW should share identical `bc` and `bq` values. The tags are also inter-dependent, so if a subread has the `bc` tag, it must also have a `bq` tag and vice-versa. If the tags are present for any subread in a ZMW, they must be present for all of them. In the absence of barcodes, both the `bc` and `bq` tags will be absent
- The `bc` tag contains the *barcode call*, a `uint16[2]` representing the inferred forward and reverse barcode sequences (as determined by their ordering in the Barcode FASTA), or more succinctly, it contains the integer pair B_F, B_R . Integer codes represent 0-based position in the FASTA file of barcodes.
- The integer (`int`) `bq` tag contains the barcode call confidence. If the `BarcodeQuality` element of the header is set to `Score`, then the tag represents the mean normalized sum of the calculated Smith-Waterman scores that support the call in the `bc` tag across all subreads. For each barcode, the sum of the Smith-Waterman score is normalized by the length of the barcode times the match score, then multiplied by 100 and rounded; this provides an integer value between 0 - 100. On the other hand, if the value of the header-tag is `Probability` instead, then the tag value is a the Phred-scaled posterior probability that the barcode call in `bc` is correct. In both cases, the value will never exceed the `int8` range, but for backward-compatibility reasons we keep the BAM `bq` as `int`. This contract allows the PBI to store `bq` as a much smaller `int8`.

The following (optional) tags describe clipped barcode sequences:

Tag	Type	Description
bl	Z	Barcode sequence clipped from leading end
bt	Z	Barcode sequence clipped from trailing end
ql	Z	Qualities of barcode bases clipped from leading end, stored as a FASTQ string
qt	Z	Qualities of barcode bases clipped from trailing end, stored as a FASTQ string
bx	B,i	Pair of clipped barcode sequence lengths

Barcode information will follow the same convention in CCS output (`ccs.bam` files).

2.17.1 Examples (subreads)

Scenario	bc	bq	cx
No barcodes, end-to-end, unknown orientation	<i>absent</i>	<i>absent</i>	1 2 = 3
Asymmetric barcodes, end-to-end, forward pass	{ 1, 37 }	35	1 2 4 8 16 = 31
Symmetric barcodes, end-to end	{ 8, 8 }	33	1 2 4 8 = 15
Barcoded, HQ region terminates before second barcode; unknown orientation	{ 8, 8 }	33	1 4 = 5

2.18 Alignment: the contract for a mapper

An aligner is expected to accept BAM input and produce aligned BAM output, where each aligned BAM record in the output preserves intact all tags present in the original record. The aligner should not attempt to orient or complement any of the tags.

(Note that this contrasts with the handling of *SEQ* and *QUAL*, which are mandated by the BAM/SAM specification to be (respectively) reverse-complemented, and reversed, for reverse strand alignments.)

2.19 Alignment: soft-clipping

In the standard production configuration, PacBio’s aligners will be used to align either subreads or CCS reads. In either case, we will use *soft clipping* to preserve the unaligned bases at either end of the query in the aligned BAM file.

2.20 Encoding of kinetics pulse features

Interpulse duration (IPD) and pulsewidth are measured in frames; natively they are recorded as a `uint16` per pulse/base event. They may be encoded in BAM read tags in one of two fashions:

- losslessly as an array of `uint16`; necessary for PacBio-internal applications but entails greater disk space usage.
- lossy 8-bit compression stored as a `uint8` array, following the codec specified below (“codec V1”). Provides a substantial disk-space savings without affecting important production use cases (base modification detection).

In the default production instrument configuration, the lossy encoding will be used. The instrument can be switched into a mode (PacBio-internal mode) where it will emit the full lossless kinetic features.

The lossy encoding for IPD and pulsewidth values into the available 256 codepoints is as follows (**codec v1**):

Frames	Encoding
0 .. 63	0, 1, .. 63
64, 66, .. 190	64, 65, .. 127
192, 196 .. 444	128, 129 .. 191
448, 456, .. 952	192, 193 .. 255

In other words, we use the first 64 codepoints to encode frame counts at single frame resolution, the next 64 to encode the frame counts at two-frame resolution, and so on. Durations exceeding 952 frames are capped at 952. Durations not enumerated in “Frames” above are rounded to the nearest enumerated duration then encoded. For example, a duration of 194 frames would round to 196 and then be encoded as codepoint 129.

This encoding has the following features, considered essential for internal analysis use cases:

- *Exact* frame-level resolution for small durations (up to 64 frames)
- Maximal representable duration is 9.52 seconds (at 100fps), which is reasonably far into the tail of the distributions of these metrics. Analyses of “pausing” phenomena may still need to account for this censoring.

A reference implementation of this encoding/decoding scheme can be found in *pbcore*.

2.21 Unresolved issues

- Need to move from strings to proper array types for QVs
- ‘/’ preferable to ‘.’ in “IPD:CodecV1”
- Desire for spec for shorter movienames, especially if these are ending up in QNAMEs.

CHAPTER 3

PacBio BAM index file (`bam.pbi`) format

PacBio's previous alignment file format (`cmp.h5`) contained a data table called the *alignment index* that recorded auxiliary identifying information and precomputed summary statistics per aligned read. This table served several purposes:

1. it enabled fast random access to aligned reads satisfying fairly complex predicates, for example, reads from a specific list of ZMWs which had unambiguous mapping (`MapQV==254`), or a read with a given readname.
2. it allowed summary reports (readlength, mapped identity/accuracy, etc.) to be constructed by quick operations over the alignment index instead of loading all of the sequence reads for each analysis.

In order to provide backwards-compatibility with the APIs enabled for accessing the `cmp.h5`, we have devised a new BAM companion file, the *PacBio BAM index*, which supports the two use cases above.

3.1 Version

This is version `4.0.0` of the `bam.pbi` specification.

3.2 Changelog

[4.0.0] - 2020-10-08

- Added `nInsOps`, `nDelOps` fields to mapped data section.

[3.0.2] - 2018-09-13

- `qStart`, `qLen` for CCS records now stored as `(0,qLen)` instead of `(-1,-1)`

[3.0.1] - 2015-09-23

- Local context flags moved into the always-present “basic” data section.

[3.0.0] - 2015-07-12

- Initial publishing of specification.

3.3 Format

The format mimics the HDF5 column-based approach without requiring the additional library dependency. Most sections are laid out as a series of 1-dimensional arrays, a la HDF5 datasets. Calculating an average or maximum mapQV, for example, would simply involve a block read of the array and the relevant computation.

It enables the 2 major use cases listed above

1. Random-access queries, including:
 - by reference or genomic region
 - by read group
 - by query name
 - by ZMW
 - by barcode index
 - etc.
2. Obtain information without processing entire BAM file
 - Calculate summary statistics
 - Reverse-lookup - get information for a record, given its index

Like the associated BAM & BAI formats, PBI is compressed in the BGZF format. See [SAM/BAM spec](#) for details.

All multi-byte numbers in PBI are stored little-endian.

All optional columns in PBI are either present for all rows or for none of them, and always present in the order described in this document. This is important for correctly accessing specific values by column index.

File sections follow each other immediately in the file and are described below.

- *PBI Header*
- *Basic Data*
- *Mapped Data* (optional)
- *Coordinate-Sorted Data* (optional)
- *Barcode Data* (optional)

3.3.1 PBI Header

Field	Size	Definition	Value
magic	char[4]	PBI magic string	PBI\1
version	uint32_t	PBI format version (xx.yy.zz)	0x00xyyyzz
pbi_flags	uint16_t	bitflag describing file contents ¹	
n_reads	uint32_t	number of reads in the BAM file	
reserved	char[18]	reserved space for future expansion	fill(0x00)

¹ pbi_flags:

Flag	Value	Description
Basic	0x0000	PbiHeader & BasicData only
Mapped	0x0001	MappedData section present
Coordinate Sorted	0x0002	CoordinateSortedData section present
Barcode	0x0004	BarcodeData section present

(0x0008 - 0x8000) are available to mark future data modifiers, add'l sections, etc.

3.3.2 Basic Data

This section contains data that apply to all PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N rgId values, followed by N qStart values, then N qEnd values, and so on.

Field	Size	Definition
rgId	int32_t	Integral value of @RG::ID ¹
qStart	int32_t	Start of query in polymerase read (0 for CCS reads)
qEnd	int32_t	End of query in polymerase read (qLen for CCS reads)
holeNumber	int32_t	The holenummer of the ZMW producing the read
readQual	float	Expected accuracy ('rq' tag) [0-1]
ctxt_flag	uint8_t	Local context of subread ('cx' tag) ²
fileOffset	int64_t	Virtual offset of record (bgzf_tell)

¹ Read group identifiers for PacBio data are calculated as follows:

```
RGID_STRING := md5(movieName + "/" + readType)) [:8]
RGID_INT    := int32.Parse(RGID_STRING)

RGID_STRING is used in the @RG header and in the `RG` tag of BAM
records.  RGID_INT is used here in the PBI index.

Note that RGID_INT may be negative.
```

² Local context flags are only valid for Subread / Insert records. For all other record-types, or if the CX tag is not present in the record, this value should be 0

3.3.3 Mapped Data

This section contains data that apply to all mapped PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N tId values, followed by N tStart values, then N tEnd values, and so on.

Field	Size	Definition
tId	int32_t	BAM tid indication aligned reference
tStart	uint32_t	(0-based) Start of alignment in reference
tEnd	uint32_t	End of alignment in reference (endpos)
aStart	uint32_t	Start of aligned query in polymerase read
aEnd	uint32_t	End of aligned query in polymerase read
revStrand	uint8_t	1 if reverse strand alignment, else 0
nM	uint32_t	Number of base matches in alignment
nMM	uint32_t	Number of base mismatches in alignment
mapQV	uint8_t	The mapping quality [valid ranges 0-254]
nInsOps	uint32_t	Number of insertion operations (not bases)
nDelOps	uint32_t	Number of deletion operations (not bases)

Note: Inserted and deleted base counts are not included in the index. These values are readily computed as:

```
nInsertedBases = aEnd - aStart - nM - nMM
nDeletedBases = tEnd - tStart - nM - nMM
```

Alignment length can be computed using:

```
aEnd - aStart + tEnd - tStart - nM - nMM
```

3.3.4 Coordinate-Sorted Data

In a coordinate-sorted BAM file, the records that are mapped to each reference form contiguous blocks. The data in this section provide a mapping between each tId and its start/end rows ².

The lookup table is prefixed with the number of reference entries.

Field	Size	Definition
n_tids	uint32_t	Number of reference sequences

The lookup table is laid out as a column (array) of tuples, one per reference.

Field	Size	Definition
tId	uint32_t	reference sequence ID ¹
beginRow	uint32_t	index of first record mapped on tId ²
endRow	uint32_t	index of last record mapped on tId ²

¹ This dataset should be sorted in *ascending order of the uint32 cast of tId* (thus a tId of -1 will follow all other tId values)

² Data fields beginRow and endRow. If $tId[i] == t$, then $[beginRow, endRow)$ represents range of reads (by 0-based ordinal position in the BAM file) mapped to the reference contig with tId of t . If no BAM records are aligned to t , then we should have $beginRow, endRow = -1$.

3.3.5 Barcode Data

This section contains data that apply to all barcoded PacBio BAM reads.

Each index field is laid out as a column (array), consisting of one value per BAM record. Thus for N records, this section will contain N bc_forward values, followed by N bc_reverse values, then N bc_qual values.

Field	Size	Definition
bc_forward	int16_t	B_F from 'bc' tag (index to barcode FASTA), or -1 if not present
bc_reverse	int16_t	B_R from 'bc' tag (index to barcode FASTA), or -1 if not present
bc_qual	int8_t	barcode call confidence ('bq' tag), or -1 if not present

Note:

If the Barcode flag is set in the header, these values must be present in all rows, otherwise it should be present for none of them.

If one Barcode field is set to -1 / non-existent, then all barcode-related fields should be set as such.

CHAPTER 4

FASTA file format

4.1 Best practices

The ubiquitous **FASTA** format is flexible, to a fault. The following best practices will guarantee success in using FASTA files with PacBio software (for example as genome references).

1. Sequences in FASTA files should be wrapped at a uniform line length, to enable indexing. (A common convention is to wrap lines at 60 characters.) Windows and UNIX line endings are both acceptable.
2. FASTA files should contain no empty lines (or lines containing only whitespace¹). FASTA files should contain no unnecessary whitespace (for example, no trailing whitespace on sequence lines).
3. Sequence contigs in a FASTA file are preceded by identifying header lines. The *header*, which is the text between the ‘>’ character and the newline, is comprised of:
 - an *identifier*, which is the first whitespace-delimited token of the header
 - an (optional) *comment*, which consists of the suffix of the header following the first whitespace. Some APIs may call this string the *description* or the *metadata*, but the usage of the comment is completely application-defined.

Sequence contigs will be identified in the PacBio system and in downstream analysis files using the *identifier*, so the *identifiers* contained in a FASTA file must be unique.

4. Sequences should only contain characters from the following string (**IUPAC** nucleotide characters, minus ‘-’ and ‘.’):

```
"gatcuryswkmbdhvnGATCURYSWKMBDHVN"
```

5. Headers should not contain any instances of the ‘>’ character, and *identifiers* should not begin with an asterisk (*) or contain any of the following characters:

```
',' ':' ' ' (double quote)
```

¹ We define a whitespace character as one for which the standard C *isspace* function returns *true* (nonzero).

6. PacBio software that imports reference genomes is allowed to outright reject FASTA files that do not meet our requirements; it will never attempt to rewrite/translate a file to fit the requirements.

4.2 Examples

The following FASTA file would be accepted by the PacBio reference uploader. Downstream files (reports, variant call files) would report the chromosomes as ‘chr1’, ‘chr2’:

```
>chr1 Jackalope chromosome 1;length=7
GATTACA
>chr2 Jackalope chromosome 2;length=7
TTACAGA
```

The following file would be *rejected* by the PacBio reference uploader, for violating the identifier uniqueness requirement (in 3, above):

```
>Jackalope chromosome 1;length=7
GATTACA
>Jackalope chromosome 2;length=7
TTACAGA
```

4.3 Implications for bioinformatics tool writers

Since a FASTA presented to a bioinformatics tool may contain characters beyond “GATCN”, tools should use a FASTA reader API capable of normalizing sequence characters. For example, [SeqAn](#) uses a *Dna5String* abstraction to project input sequence characters into “GATCN”; FASTA readers in pbcore will soon offer normalization.

CHAPTER 5

DataSet format specification

A PacBio DataSet is an XML file representing a set of a particular sequence data type such as subreads, references or aligned subreads. The actual data elements contained in a DataSet are stored in files referred to by the XML, usually in FASTA or BAM files. The DataSet can optionally filter these files or store metadata about their contents.

5.1 Version

This document defines the 4.0.1 SMRT Link DataSet abstraction and its XML file representation. The 4.0.1 format is produced by the 11.0.0 release of PacBio Primary Analysis and SMRT Link software.

5.2 Motivating Use Cases

The concept of a homogenous set of elements of a particular data type is used throughout Pacific Biosciences software. These “sets of data” have historically been represented in different ways, sometimes explicitly by creating large files that contain all the data in the set (e.g. a FASTA file of subreads), and sometimes implicitly by using pointers to the data, such as with a FOFN (file of file names).

In many cases these solutions are serviceable, but they have disadvantages. The reliance on explicit file creation imposes a heavy burden that will be exacerbated by future instrument throughput increases. The FOFN partially breaks the tight coupling between explicit files and sets of data, but it fails to allow facile subsetting of files. With previous methods, consuming tools were forced to reimplement filtering or subsetting logic in their own idiosyncratic ways.

The DataSet XML abstraction (or DataSet for short) addresses these problems by providing a standard way to perform the following activities:

- Refer to a **set of subreads** in multiple BAM files
- Refer to a **subset of subreads** by id from one or more BAM files
- Refer to a **subset of alignments** for input to algorithms such as Arrow (e.g. on a particular chromosome, or a particular chromosome region)

- Perform any analysis that can be performed on an entire file of a particular data type (subreads, alignments) on a **subset of that data type** without creating a new file.

5.3 Format Definition

5.3.1 XML Representation

The canonical representation of a DataSet is an XML file that contains a single DataSet element with four major sections, one mandatory and three optional:

1. A mandatory `<pbbase:ExternalResources>` section with references to external data sources in BAM or FASTA format and their associated indices or metadata files. The records in these files are the elements of the set of data represented by the DataSet.
2. An optional `<pbdbs:Filters>` section that filters or subsets the elements of the set in the above files, for example by length.
3. An optional `<pbdbs:DataSetMetadata>` section that contains metadata about the DataSet, usually at minimum the number of records and their total length, but possibly much more. For example, subread and CCS read DataSets have metadata regarding instrument collection or biological samples. The Metadata section should be considered to refer to the DataSet elements prior to applying Filters.
4. An optional `<pbdbs:DataSets>` section that can be used to annotate subsets of the top-level DataSet. For example, a DataSet that resulted from merging multiple DataSets can store a record of the original DataSets in this section, along with their original metadata.
5. An optional `<pbbase:SupplementalResources>` section with references to external data or metadata sources in any format, separate from the main data files in ExternalResources, but represented using the same `<pbbase:ExternalResource>` model. This may include files such as reports or unbarcoded reads BAM.

5.3.2 Types of DataSets

The DataSet `XSD` defines DataSet subclasses for the most common entities consumed and produced by SMRT Analysis pipelines:

- **SubreadSet** - This type of DataSet represents the basic sequence data generated by the Sequel instrument, with `ExternalResource`'s in BAM format. Note that the prefix "Sub" in this case is part of "Subread", and does not in any way denote a "subset".
- **ConsensusReadSet** - CCS sequence data in BAM format.
- **AlignmentSet** - Aligned reads in BAM format.
- **ConsensusAlignmentSet** - Aligned CCS data in BAM format.
- **ReferenceSet** - The FASTA reference and associated indices used by Resequencing and Base Mods. Replaces the reference repository entries.
- **GmapReferenceSet** - A FASTA ReferenceSet with a GMAP database.
- **ContigSet** - Any FASTA containing contigs, e.g. those produced by HGAP.
- **BarcodeSet** - The FASTA file and metadata used by barcode detection.
- **TranscriptSet** - Processed RNA transcripts in BAM format.
- **TranscriptAlignmentSet** - Processed and aligned RNA transcripts in BAM format.

5.3.3 SubreadSet example

Here is a simple example of a DataSet XML using a SubreadSet containing all four sections. It creates a set of subreads from two subread BAM files and associated indices and filters the subreads by quality using the `rq` field of the underlying BAM records:

```
<?xml version="1.0" encoding="utf-8"?>
<pbd:SubreadSet
  xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbdbs="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="subreadset_150304_231155"
  MetaType="PacBio.DataSet.SubreadSet"
  Name="DataSet_SubreadSet"
  Tags=""
  Version="4.0.1"
  CreatedAt="2015-01-27T09:00:01">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
      TimeStampedName="subread_bam_150304_231155"
      MetaType="PacBio.SubreadFile.SubreadBamFile"
      ResourceId="m150404_101626_42267_s1_p0.1.subreads.bam">
      <pbbase:FileIndices>
        <pbbase:FileIndex
          UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5194"
          TimeStampedName="bam_index_150304_231155"
          MetaType="PacBio.Index.PacBioIndex"
          ResourceId="m150404_101626_42267_s1_p0.1.subreads.bam.pbi"/>
        </pbbase:FileIndices>
      </pbbase:ExternalResource>
      <pbbase:ExternalResource
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5197"
        TimeStampedName="subread_bam_150304_231155"
        MetaType="PacBio.SubreadFile.SubreadBamFile"
        ResourceId="m150404_101626_42267_s1_p0.2.subreads.bam">
        <pbbase:FileIndices>
          <pbbase:FileIndex
            UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5198"
            TimeStampedName="bam_index_150304_231155"
            MetaType="PacBio.Index.PacBioIndex"
            ResourceId="m150404_101626_42267_s1_p0.2.subreads.bam.pbi"/>
          </pbbase:FileIndices>
        </pbbase:ExternalResource>
      </pbbase:ExternalResources>
    <pbd:Filters>
      <pbd:Filter>
        <pbbase:Properties>
          <pbbase:Property Name="rq" Operator="gt" Value="0.80"/>
        </pbbase:Properties>
      </pbd:Filter>
    </pbd:Filters>
  <pbd:DataSetMetadata>
```

(continues on next page)

(continued from previous page)

```

    <pbbase:TotalLength>5000</pbbase:TotalLength>
    <pbbase:NumRecords>500</pbbase:NumRecords>
  </pbds:DataSetMetadata>
</pbds:SubreadSet>

```

5.3.4 Operations on DataSets

DataSets support operations that would naively be expected of sets, such as subsetting and union (although notably not intersection) as well as some additional operations such as consolidation. The result of performing these operations is itself a new DataSet, with the operations included as a kind of “recipe” for producing the new DataSet from the original. Because of this, operations are presented here as part of the DataSet format.

Subsetting (Filtering)

The SubreadSet example given above is the result of applying a length subset or `Filter` operation on a DataSet of two BAM files (e.g. by applying the `dataset` command).

Each `Filter` is composed of `Property` tags representing logical predicates that elements of the DataSet must satisfy. `Property` tags are defined by three attributes: `Name`, `Operator` and `Value`, where the `Name` refers to a field or derived value from individual DataSet records, and the `Operator` is used to compare a particular record’s field with the `Value` attribute. Individual `Filter` tags can be combined to create more complicated filters; these `Filter` tags are logically “ORed” while individual `Property` tags within a `Filter` are “ANDed” together.

In summary, the `Filter` and `Property` tags provide a powerful means of subsetting DataSets without manipulating the underlying file representations.

Filter Property Names and Values

The following table shows `Property Name`s that are supported by either the pbbam C++ API, the pbcore Python API (*italicized*) or both (**bold**). Some accepted alternative ways of representing a `Property` are given in the `Alternative Names` column. Those `Property`s that allow list values in both pbbam and pbcore are indicated by brackets. Finally, the right column shows the allowed type of `Property Value` associated with that particular `Property Name`.

Property Name	Description	Alternative Names	Value
readstart	Alignment start	astart as	uint32_t
ae	Alignment end	aend	uint32_t
alignedlength	Alignment length		uint32_t
accuracy	Alignment identity	identity	float
qname	Query name	<i>qid</i>	string
qstart	Query start	qs	int
qend	Query end	qe	int
length	Query length	querylength	int
rname	Reference name		string
tstart	Reference start	ts pos	uint32_t
tend	Reference end	te	uint32_t
qname_file	Query names from a file		filename
rq	Predicted read quality		float
movie	Movie name (e.g. m150404_101626_42267_s1_p0)		string
zm	ZMW (e.g. m150404_101626_42267_s1_p0/100)	zmw	string[]
bc	Barcode name	barcode	string[]
bcq	Barcode quality	bq	uint8_t
bcf	Barcode forward		string[]
bcr	Barcode reverse		string[]
cx	Local context (see below for special Values)		see below
<i>n_subreads</i>	Number of subreads		int
<i>mapqv</i>	Mapping quality		int

Filter Property Operators

The following table shows supported Property Operator's:

Property Operator	Description	Other Names
==	Equal to	= eq
!=	Not Equal to	ne
<	Less Than	lt <
<=	Less than or equal to	lte <=
>	Greater than	gt >
>=	Greater than or equal to	gte >=
&	Contains	and
~	Does not contain	not

Possible cx Values

The following table shows Values's that are supported for the **cx** local context Property Name. The Value's can be used individually or can be combined to form a compound Value using a syntax that looks and behaves similarly to OR-ing bitflags, e.g.:

```
Value='ADAPTER_BEFORE | ADAPTER_AFTER'
```

cx Value	Description
NO_LOCAL_CONTEXT	No local context (e.g. adapters or barcodes)
ADAPTER_BEFORE	An adapter was detected before (5' to) the subread
ADAPTER_AFTER	An adapter was detected after (3' to) this subread
BARCODE_BEFORE	A barcode was detected before (5' to) this subread
BARCODE_AFTER	A barcode was detected after (3' to) this subread
FORWARD_PASS	Subread is forward on the polymerase read
REVERSE_PASS	Subread is forward on the polymerase read

Filtering and the pbi

The subsetting/filtering operation is supported efficiently by the PacBio index (*.pbi files). The contents of DataSet after applying `Filter`'s should in general be decidable using the contents of the pbi file (`rname` is the one exception in 4.0 and requires examination of the BAM file header).

Union (Merging)

Unions can be taken of DataSets with the same underlying file type (noted by the `MetaType` attribute) and with identical `Filter`'s. The output of a union (or merge) operation is a *single* new DataSet XML file containing a *single* new top-level DataSet element—notably not multiple top-level elements. For example, the SubreadSet above could be created by taking the union of two SubreadSets each containing a single BAM file:

```
<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="rq" Operator="gt" Value="0.75"/>
      </pbbase:Properties>
    </pbds:Filter>
  </pbds:Filters>
  <pbbase:DataSetMetadata>
    <pbbase:TotalLength>3000</pbbase:TotalLength>
    <pbbase:NumRecords>300</pbbase:NumRecords>
  </pbbase:DataSetMetadata>
</pbds:SubreadSet>

<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads1.bam"/>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="rq" Operator="gt" Value="0.75"/>
      </pbbase:Properties>
    </pbds:Filter>
  </pbds:Filters>
```

(continues on next page)

(continued from previous page)

```

<pbd:DataSetMetadata>
  <pbbase:TotalLength>2000</pbbase:TotalLength>
  <pbbase:NumRecords>200</pbbase:NumRecords>
</pbd:DataSetMetadata>
</pbd:SubreadSet>

```

Tools that merge multiple DataSets together can optionally store the original input DataSets in the DataSets tag of the output DataSet. In this way the original DataSets are maintained as “subdatasets” of the new DataSet. Merging tools should adhere to the following best practices:

- Subdatasets should be created for each input DataSet when two or more files with no subdatasets are merged or used to create a new DataSet.
- If in the process of merging a new DataSet is created, for example, wrapping a “naked” BAM file, a subdataset should be created for those inputs.
- In no case should a subdataset be present if it is identical to the containing DataSet.
- Subdatasets should be preserved when two datasets with subdatasets are merged together.
- If both have subdatasets, the lists of subdatasets should be concatenated.
- If one has no subdatasets, a subdataset should be created for that input and added to the list of subdatasets from the other input file.
- Subdatasets should be preserved during minor manipulations: Adding a filter, changing path absoluteness, adding or removing indices, references etc.
- Subdatasets should be removed during substantial transformations: alignment, producing an AlignmentSet from a SubreadSet.
- Subdatasets should be preserved during consolidation or splitting (except when splitting by subdataset).

Consolidating

Consolidating (aka Resolving) a DataSet means creating an explicit representation in the appropriate format with all filters applied. In practice, this means building a single BAM file containing all the records implied by the DataSet’s ExternalResource and Filter directives. Here is consolidated version of the SubreadSet above:

```

<?xml version="1.0" encoding="utf-8" ?>
<pbd:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0_plus_subreads1.bam"/>
  </pbbase:ExternalResources>
  <pbd:DataSetMetadata>
    <pbbase:TotalLength>5000</pbbase:TotalLength>
    <pbbase:NumRecords>500</pbbase:NumRecords>
  </pbd:DataSetMetadata>
</pbd:SubreadSet>

```

Consolidated DataSets are useful for export of a DataSet that exists only implicitly, e.g. by filtering multiple files. They allow the user to incur the IO overhead of seeking over multiple files once at the cost of increased disk usage.

5.3.5 DataSet MetaTypes and File Extensions

The following table lists the DataSet subclasses defined by the XSD and their associated MetaType values and expected filename suffix strings. While technically the information conveyed by the MetaType and suffix is redundant, it is useful for some downstream consuming tools, and to be fully compliant with this specification all three should be consistent.

DataSet	DataSet MetaType	DataSet XML File Extension
SubreadSet	PacBio.DataSet.SubreadSet	.subreadset.xml
HdfSubreadSet	PacBio.DataSet.HdfSubreadSet	.hdfsubreadset.xml
AlignmentSet	PacBio.DataSet.AlignmentSet	.alignmentset.xml
BarcodeSet	PacBio.DataSet.BarcodeSet	.barcodeset.xml
ConsensusReadSet	PacBio.DataSet.ConsensusReadSet	.consensusreadset.xml
ConsensusAlignmentSet	PacBio.DataSet.ConsensusAlignmentSet	.consensusalignmentset.xml
ContigSet	PacBio.DataSet.ContigSet	.contigset.xml
ReferenceSet	PacBio.DataSet.ReferenceSet	.referenceset.xml
GmapReferenceSet	PacBio.DataSet.GmapReferenceSet	.gmapreferenceset.xml
TranscriptSet	PacBio.DataSet.TranscriptSet	.transcriptset.xml
TranscriptAlignmentSet PacBio.DataSet.TranscriptAlignmentSet .transcriptalignmentset.xml		

DataSet External Resource MetaTypes and File Extensions

The following table lists the ExternalResource's expected by each DataSet subclass and their associated MetaType values and expected filename suffix strings. In some cases the ExternalResource can occur in multiple DataSet types, but only for particular parent ExternalResource's elements. Note that while this specification allows ExternalResource's to in some cases refer to other DataSets, they should in no cases refer to a DataSet of the same type as the parent (e.g. SubreadSet's should not refer to external SubreadSet's).

DataSet or Tag	ExternalResource MetaType	File Extensions
SubreadSet	PacBio.SubreadFile.SubreadBamFile	.bam
	PacBio.SubreadFile.ScrapBamFile	.bam
	PacBio.SubreadFile.HqRegionBamFile	.bam
	PacBio.SubreadFile.HqScrapBamFile	.bam
	PacBio.SubreadFile.LqRegionBamFile	.bam
	PacBio.SubreadFile.LqScrapBamFile	.bam
	PacBio.SubreadFile.PolymeraseBamFile	.bam
	PacBio.SubreadFile.PolymeraseScrapBamFile	.bam
	PacBio.SubreadFile.ChipStatsFile	.sts.xml
	PacBio.SubreadFile.ChipStatsH5File	.sts.h5
	PacBio.SubreadFile.AdapterFastaFile	.fasta
	PacBio.SubreadFile.ControlFastaFile	.fasta
HdfSubreadSet	PacBio.SubreadFile.BaxFile	.bax.h5
AlignmentSet	PacBio.AlignmentFile.AlignmentBamFile	.bam
BarcodeSet	PacBio.BarcodeFile.BarcodeFastaFile	.fasta
ConsensusReadSet	PacBio.ConsensusReadFile.ConsensusReadBamFile	.bam
ConsensusAlignmentSet	PacBio.AlignmentFile.ConsensusAlignmentBamFile	.bam
TranscriptSet	PacBio.TranscriptFile.TranscriptBamFile	.bam
TranscriptAlignmentSet	PacBio.AlignmentFile.TranscriptAlignmentBamFile	.bam
ContigSet	PacBio.ContigFile.ContigFastaFile	.fasta
ReferenceSet	PacBio.ReferenceFile.ReferenceFastaFile	.fasta
Bam ExternalResource	PacBio.Index.BamIndex	.bam.bai
	PacBio.Index.PacBioIndex	.bam.pbi
Fasta ExternalResource	PacBio.Index.SamIndex	.fasta.fai
	PacBio.Index.SaWriterIndex	.fasta.sa
	PacBio.GmapDB.GmapDBSummary	?
	PacBio.Index.NgmlrRefEncoded	.ngm
	PacBio.Index.NgmlrRefTable	.ngm

Some `ExternalResource`'s themselves contain associated `ExternalResource`'s, for example the indices associated with FASTA files. These associated files are nested within the primary `ExternalResource` element to denote their subsidiary nature.

DataSet UI Name and Time Stamped Name

The pattern for `TimeStampedName` attribute generated by production PacBio tools should be:

```
<metatype>-<yymmdd_HHmmsstt>
```

Where `MetaType` has been transformed into a lowercase, underscore separated string and the time string format directives map to the following entities: year, month, day, hour, minute, second, millisecond.

DataSet	TimeStampedName	DataSet UI Name
SubreadSet	pacbio_dataset_subreadset-<yymmdd_HHmmssttt>	BAM Data
HdfSubreadSet	pacbio_dataset_hdfsubreadset-<yymmdd_HHmmssttt>	RS II Data
AlignmentSet	pacbio_dataset_alignmentset-<yymmdd_HHmmssttt>	TBD
BarcodeSet	pacbio_dataset_barcodeaset-<yymmdd_HHmmssttt>	Barcode
ConsensusReadSet	pacbio_dataset_consensusreadset-<yymmdd_HHmmssttt>	TBD
ConsensusAlign- mentSet	pacbio_dataset_consensusalignmentset- <yymmdd_HHmmssttt>	TBD
ContigSet	pacbio_dataset_contigset-<yymmdd_HHmmssttt>	TBD
ReferenceSet	pacbio_dataset_referenceset-<yymmdd_HHmmssttt>	Reference
TranscriptSet	pacbio_dataset_transcriptset-<yymmdd_HHmmssttt>	TBD
TranscriptAlignmentSet	pacbio_dataset_transcriptalignmentset-<yymmdd_HHmmssttt>	TBD

Note that all PacBio.Index.*ExternalResource's are given a TimeStampedName attribute with an "index-" prefix.

5.3.6 Singleton DataSet element

While technically an XML file with multiple DataSet elements is valid under the XSD and this spec, in practice PacBio tools follow a one DataSet element per XML file convention.

5.4 Support for the DataSet XML

Support exists for using the DataSet XML throughout the SMRT Analysis stack.

5.4.1 The dataset command-line tool

The SMRT Link CL tools in 10.0.0 provide command-line support for creating, filtering, validating, merging, splitting, consolidating and other common operations via the `dataset` command. Detailed CL documentation and example can be found in the [pbcoretools](#) documentation. Note that in the SMRT Link CL tools tarball, the command is `dataset` not `dataset.py`.

5.4.2 Core API Support

An API is provided in [pbcore](#) (Python) and [pbbam](#) (C++) that makes consuming DataSet XML files or the underlying files such as BAM as burden-free as possible.

5.4.3 Other PacBio command-line tools

All bioinformatics tools that consume DataSet XML files should be capable of producing identical results using the equivalent BAM or FASTA file generated after applying all filters. In other words, the DataSet XML is not required to obtain bioinformatics results. However, *support* for the DataSet XML is encouraged for SMRT Analysis tools, and facilitated using the above API.

5.4.4 SMRT Link and SMRT Pipe

The DataSet XML is required for display of DataSets (such as references) in SMRT Link and for chunking in the distributed pipelines using pbsmrtpipe. Moreover, for these applications the DataSetMetadata field is mandatory, not optional.

5.5 Other topics

5.5.1 Mutability and equality

To allow user editing of attributes such as Name without affecting the underlying DataSet we define the Core DataSet as the XML with the user editable attributes (Name, Description and Tags) removed (not set to "", but absent). This Core DataSet is immutable and is the entity on which identity operations are defined. As a consequence, any modifications to fields other than Name, Description or Tags requires giving the DataSet a new universal UniqueId (aka UUID). Operations such as md5 checksum should be performed on the Core DataSet unless otherwise specified.

5.5.2 I/O trade-offs

The DataSet model defers I/O operations by replacing up-front file merges with downstream I/O operations that hit many different files. This allows consumers to avoid explicit creation of files on disk and the resulting redundant storage and costly write operations. For many uses this many-file approach is preferred to explicitly creating the file on disk, but in some cases it may be desirable to incur the cost of accessing and filtering multiple files once (e.g. to reduce disk seeks for highly fragmented DataSets). Determining when the costs outweigh the benefits will need practical investigation, but regardless the Consolidate operation provides the means for using the form of DataSet that best fits a particular use case.

5.5.3 Outstanding Issues and Future Directions

- Document FASTA filters for pbcore / pbbam for releases post-4.0.
- The propagation of subdatasets in merging can result in rather large XML files with duplicated information. It is possible this duplication could be reduced using XML IDREFs from the subdatasets to information in the top level DataSet, for ExternalResources or CollectionMetadata. This should be considered as a possible future revision.

5.6 Appendix 1: Examples satisfying the motivating use cases

- Refer to a **set of subreads** in multiple BAM files:

```
<?xml version="1.0" encoding="utf-8" ?>
<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads1.bam"/>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
```

(continues on next page)

(continued from previous page)

```

        <pbbase:Properties>
          <pbbase:Property Name="rq" Operator="gt" Value="0.75"/>
        </pbbase:Properties>
      </pbds:Filter>
    </pbds:Filters>
    <pbds:DataSetMetadata>
      <pbbase:TotalLength>5000</pbbase:TotalLength>
      <pbbase:NumRecords>500</pbbase:NumRecords>
    </pbds:DataSetMetadata>
  </pbds:SubreadSet>

```

- Refer to a **subset of subreads** by id from one or more BAM files:

```

<?xml version="1.0" encoding="utf-8" ?>
<pbds:SubreadSet xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource MetaType="SubreadFile.SubreadBamFile"
      ResourceId="file:/mnt/path/to/subreads0.bam"/>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="qname" Operator="==" Value="m100000/0/0_100
→"/>
      </pbbase:Properties>
    </pbds:Filter>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="qname" Operator="==" Value="m100000/1/0_200
→"/>
      </pbbase:Properties>
    </pbds:Filter>
  </pbds:Filters>
  <pbds:DataSetMetadata>
    <pbbase:TotalLength>5000</pbbase:TotalLength>
    <pbbase:NumRecords>500</pbbase:NumRecords>
  </pbds:DataSetMetadata>
</pbds:SubreadSet>

```

- Refer to a **subset of alignments** for input to algorithms such as Arrow (e.g. on a particular chromosome, or a particular ch

- Use an *AlignmentSet* filtered by *rname*

- Perform any analysis that can be performed on an entire file of a particular data type (reads, read regions, alignments) on a **subset of that data type** without creating a new file. - *Relies on tools using common APIs to access DataSets.*

5.7 Appendix 2: Example DataSet XML files

Here are some example XML files for each of the above DataSets

Example SubreadSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:SubreadSet
  xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="subreadset_150304_231155"
  MetaType="PacBio.DataSet.SubreadSet"
  Name="DataSet_SubreadSet"
  Tags="barcode moreTags mapping mytags"
  Version="3.0.0"
  CreatedAt="2015-01-27T09:00:01">
<pbbase:ExternalResources>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
    TimeStampedName="subread_bam_150304_231155"
    MetaType="PacBio.SubreadFile.SubreadBamFile"
    Name="First Subreads BAM"
    Description="Points to an example Subreads BAM file."
    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5194"
        TimeStampedName="bam_index_150304_231155"
        MetaType="PacBio.Index.PacBioIndex"
        ResourceId="file:///mnt/path/to/subreads0.pbi"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResource>
    <pbbase:ExternalResource
      UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5195"
      TimeStampedName="scraps_bam_150304_231155"
      MetaType="PacBio.SubreadFile.ScrapsBamFile"
      ResourceId="file:///mnt/path/to/scraps0.bam"
      Tags="Example"
      Name="First scraps BAM"
      Description="Points to an example scraps BAM file.">
      <pbbase:FileIndices>
        <pbbase:FileIndex
          UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5196"
          TimeStampedName="bam_index_150304_231155"
          MetaType="PacBio.Index.PacBioIndex"
          ResourceId="file:///mnt/path/to/scraps0.pbi"/>
        </pbbase:FileIndices>
      </pbbase:ExternalResource>
    </pbbase:ExternalResources>
  </pbbase:ExternalResource>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5197"
    TimeStampedName="subread_bam_150304_231155"
    MetaType="PacBio.SubreadFile.SubreadBamFile"
    ResourceId="file:///mnt/path/to/subreads1.bam"
    Name="Second Subreads BAM"
    Description="Points to another example Subreads BAM file."

```

(continues on next page)

(continued from previous page)

```

    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b096d0a3-94b8-4918-b3af-a3f81bbe5198"
        TimeStampedName="bam_index_150304_231155"
        MetaType="PacBio.Index.PacBioIndex"
        ResourceId="file:///mnt/path/to/subreads0.pbi"/>
      </pbbase:FileIndices>
    <pbbase:ExternalResources>
      <pbbase:ExternalResource
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5199"
        TimeStampedName="scraps_bam_150304_231155"
        MetaType="PacBio.SubreadFile.ScrapsBamFile"
        ResourceId="file:///mnt/path/to/scraps1.bam"
        Name="Second scraps BAM" Description="Points to another example_
↳scraps BAM file."
        Tags="Example">
        <pbbase:FileIndices>
          <pbbase:FileIndex
            UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5190"
            TimeStampedName="bam_index_150304_231155"
            MetaType="PacBio.Index.PacBioIndex"
            ResourceId="file:///mnt/path/to/scraps1.pbi"/>
          </pbbase:FileIndices>
        </pbbase:ExternalResource>
      </pbbase:ExternalResources>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
  <pbds:Filters>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="rq" Value="0.75" Operator=">"/>
      </pbbase:Properties>
    </pbds:Filter>
    <pbds:Filter>
      <pbbase:Properties>
        <pbbase:Property Name="QNAME" Value="100/0/0_100" Operator="==" />
      </pbbase:Properties>
    </pbds:Filter>
  </pbds:Filters>
  <pbds:DataSetMetadata>
    <pbds:TotalLength>500000</pbds:TotalLength>
    <pbds:NumRecords>500</pbds:NumRecords>
    <pbmeta:Collections>
      <pbmeta:CollectionMetadata
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
        TimeStampedName="collection_150304_231155"
        MetaType="PacBio.CollectionMetadata"
        Context="m152720_092723_00114_c100480560100000001823075906281381_s1_p0"
        InstrumentName="RS"
        InstrumentId="43210">
        <pbmeta:InstCtrlVer>2.3.0.0.140640</pbmeta:InstCtrlVer>
        <pbmeta:SigProcVer>NRT@172.31.128.10:8082, SwVer=2300.140640, HwVer=1.0</
↳pbmeta:SigProcVer>
        <pbmeta:RunDetails>
          <pbmeta:RunId>e903682f-e502-465c-a2b6-9dd77c9f43fc</pbmeta:RunId>
          <pbmeta:Name>beta4_130726_biotin_DEV_vs_MFG_PB11K_9458p</pbmeta:Name>

```

(continues on next page)

(continued from previous page)

```

    </pbmeta:RunDetails>
    <pbmeta:WellSample Name="Well Sample 1">
      <pbmeta:PlateId>2014-12-24_141_NGAT_Igor_bisPNA Enrichment_Mag Bead_
↪Elution Buffers</pbmeta:PlateId>
      <pbmeta:WellName>B01</pbmeta:WellName>
      <pbmeta:Concentration>10</pbmeta:Concentration>
      <pbmeta:SampleReuseEnabled>true</pbmeta:SampleReuseEnabled>
      <pbmeta:StageHotstartEnabled>true</pbmeta:StageHotstartEnabled>
      <pbmeta:SizeSelectionEnabled>true</pbmeta:SizeSelectionEnabled>
      <pbmeta:UseCount>0</pbmeta:UseCount>
      <pbsample:BioSamplePointers>
        <pbsample:BioSamplePointer>abc2df90-d44f-4a48-9f35-3b99473c68f5</
↪pbsample:BioSamplePointer>
      </pbsample:BioSamplePointers>
    </pbmeta:WellSample>
    <pbmeta:Automation>
      <pbbase:AutomationParameters>
        <pbbase:AutomationParameter/>
      </pbbase:AutomationParameters>
    </pbmeta:Automation>
    <pbmeta:CollectionNumber>0</pbmeta:CollectionNumber>
    <pbmeta:CellIndex>0</pbmeta:CellIndex>
    <pbmeta:CellPac Barcode="100480560100000001823075906281381"/>
    <pbmeta:Primary>
      <pbmeta:AutomationName>BasecallerV1</pbmeta:AutomationName>
      <pbmeta:ConfigFileName>1-3-0_Standard_C2.xml</pbmeta:ConfigFileName>
      <pbmeta:SequencingCondition/>
      <pbmeta:OutputOptions>
        <pbmeta:ResultsFolder>Analysis_Results</pbmeta:ResultsFolder>
        <pbmeta:CollectionPathUri>rsy://mp-rsync/vol56//RS_DATA_STAGING//
↪2014-12-24_141_NGAT_Igor_bisPNA%20Enrichment_Mag%20Bead%20Elution%20Buffers_1094/
↪B01_1</pbmeta:CollectionPathUri>
        <pbmeta:CopyFiles>
          <pbmeta:CollectionFileCopy>Bam</pbmeta:CollectionFileCopy>
        </pbmeta:CopyFiles>
        <pbmeta:Readout>Bases</pbmeta:Readout>
        <pbmeta:MetricsVerbosity>Minimal</pbmeta:MetricsVerbosity>
      </pbmeta:OutputOptions>
    </pbmeta:Primary>
  </pbmeta:CollectionMetadata>
</pbmeta:Collections>
<pbsample:BioSamples>
  <pbsample:BioSample
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
    TimeStampedName="biosample_150304_231155"
    MetaType="PacBio.BioSample"
    Name="consectetur purus"
    Description="Risus sit amet lectus vehicula vulputate quisque porta_
↪accumsan venenatis."
    CreatedAt="2015-01-20T13:27:23.9271737-08:00"/>
  </pbsample:BioSamples>
<pbds:SummaryStats>
  <pbds:AdapterDimerFraction>0.1</pbds:AdapterDimerFraction>
  <pbds:ShortInsertFraction>0.1</pbds:ShortInsertFraction>
  <pbds:NumSequencingZmws>0</pbds:NumSequencingZmws>
  <pbds:ProdDist>
    <pbbase:NumBins>4</pbbase:NumBins>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:BinCounts>
      <pbbase:BinCount>1576</pbbase:BinCount>
      <pbbase:BinCount>901</pbbase:BinCount>
      <pbbase:BinCount>399</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:MetricDescription>Productivity</pbbase:MetricDescription>
    <pbbase:BinLabels>
      <pbbase:BinLabel>Empty</pbbase:BinLabel>
      <pbbase:BinLabel>Productive</pbbase:BinLabel>
      <pbbase:BinLabel>Other</pbbase:BinLabel>
      <pbbase:BinLabel>NotDefined</pbbase:BinLabel>
    </pbbase:BinLabels>
  </pbd:ProdDist>
  <pbd:ReadTypeDist>
    <pbbase:NumBins>9</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>1474</pbbase:BinCount>
      <pbbase:BinCount>799</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>181</pbbase:BinCount>
      <pbbase:BinCount>92</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>326</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:MetricDescription>Read Type</pbbase:MetricDescription>
    <pbbase:BinLabels>
      <pbbase:BinLabel>Empty</pbbase:BinLabel>
      <pbbase:BinLabel>FullHqRead0</pbbase:BinLabel>
      <pbbase:BinLabel>FullHqRead1</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead0</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead1</pbbase:BinLabel>
      <pbbase:BinLabel>PartialHqRead2</pbbase:BinLabel>
      <pbbase:BinLabel>Multiload</pbbase:BinLabel>
      <pbbase:BinLabel>Indeterminate</pbbase:BinLabel>
      <pbbase:BinLabel>NotDefined</pbbase:BinLabel>
    </pbbase:BinLabels>
  </pbd:ReadTypeDist>
  <pbd:ReadLenDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>4528.69384765625</pbbase:SampleMean>
    <pbbase:SampleMed>5227</pbbase:SampleMed>
    <pbbase:SampleStd>2322.8055598026981</pbbase:SampleStd>
    <pbbase:Sample95thPct>7367</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>62</pbbase:BinCount>
      <pbbase:BinCount>39</pbbase:BinCount>
      <pbbase:BinCount>36</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>19</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>32</pbbase:BinCount>
        <pbbase:BinCount>32</pbbase:BinCount>
        <pbbase:BinCount>40</pbbase:BinCount>
        <pbbase:BinCount>45</pbbase:BinCount>
        <pbbase:BinCount>54</pbbase:BinCount>
        <pbbase:BinCount>73</pbbase:BinCount>
        <pbbase:BinCount>77</pbbase:BinCount>
        <pbbase:BinCount>97</pbbase:BinCount>
        <pbbase:BinCount>95</pbbase:BinCount>
        <pbbase:BinCount>49</pbbase:BinCount>
        <pbbase:BinCount>17</pbbase:BinCount>
        <pbbase:BinCount>2</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        </pbbase:BinCounts>
        <pbbase:BinWidth>418.89999389648438</pbbase:BinWidth>
        <pbbase:MinOutlierValue>77</pbbase:MinOutlierValue>
        <pbbase:MinBinValue>77</pbbase:MinBinValue>
        <pbbase:MaxBinValue>12225.1025390625</pbbase:MaxBinValue>
        <pbbase:MaxOutlierValue>12644</pbbase:MaxOutlierValue>
        <pbbase:MetricDescription>Polymerase Read Length</
    <pbbase:MetricDescription>
  </pbds:ReadLenDist>
  <pbds:ReadQualDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>0.82736450433731079</pbbase:SampleMean>
    <pbbase:SampleMed>0.83167940378189087</pbbase:SampleMed>
    <pbbase:SampleStd>0.029663275550147809</pbbase:SampleStd>
    <pbbase:Sample95thPct>0.86801999807357788</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>12</pbbase:BinCount>
      <pbbase:BinCount>8</pbbase:BinCount>
      <pbbase:BinCount>11</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>17</pbbase:BinCount>
      <pbbase:BinCount>11</pbbase:BinCount>
      <pbbase:BinCount>20</pbbase:BinCount>
      <pbbase:BinCount>24</pbbase:BinCount>
      <pbbase:BinCount>24</pbbase:BinCount>
      <pbbase:BinCount>34</pbbase:BinCount>
      <pbbase:BinCount>25</pbbase:BinCount>
      <pbbase:BinCount>38</pbbase:BinCount>
      <pbbase:BinCount>52</pbbase:BinCount>
      <pbbase:BinCount>33</pbbase:BinCount>
      <pbbase:BinCount>45</pbbase:BinCount>
      <pbbase:BinCount>48</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>47</pbbase:BinCount>
        <pbbase:BinCount>69</pbbase:BinCount>
        <pbbase:BinCount>65</pbbase:BinCount>
        <pbbase:BinCount>55</pbbase:BinCount>
        <pbbase:BinCount>51</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>42</pbbase:BinCount>
        <pbbase:BinCount>29</pbbase:BinCount>
        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>3</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
→pbbase:MetricDescription>
  </pbds:ReadQualDist>
  <pbds:ControlReadLenDist>
    <pbbase:SampleSize>901</pbbase:SampleSize>
    <pbbase:SampleMean>4528.69384765625</pbbase:SampleMean>
    <pbbase:SampleMed>5227</pbbase:SampleMed>
    <pbbase:SampleStd>2322.8055598026981</pbbase:SampleStd>
    <pbbase:Sample95thPct>7367</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>62</pbbase:BinCount>
      <pbbase:BinCount>39</pbbase:BinCount>
      <pbbase:BinCount>36</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>19</pbbase:BinCount>
      <pbbase:BinCount>29</pbbase:BinCount>
      <pbbase:BinCount>37</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>40</pbbase:BinCount>
      <pbbase:BinCount>45</pbbase:BinCount>
      <pbbase:BinCount>54</pbbase:BinCount>
      <pbbase:BinCount>73</pbbase:BinCount>
      <pbbase:BinCount>77</pbbase:BinCount>
      <pbbase:BinCount>97</pbbase:BinCount>
      <pbbase:BinCount>95</pbbase:BinCount>
      <pbbase:BinCount>49</pbbase:BinCount>
      <pbbase:BinCount>17</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>

```

(continues on next page)

(continued from previous page)

```

        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>418.89999389648438</pbbase:BinWidth>
    <pbbase:MinOutlierValue>77</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>77</pbbase:MinBinValue>
    <pbbase:MaxBinValue>12225.1025390625</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>12644</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Length</
    <pbbase:MetricDescription>
    </pbds:ControlReadLenDist>
    <pbds:ControlReadQualDist>
        <pbbase:SampleSize>901</pbbase:SampleSize>
        <pbbase:SampleMean>0.82736450433731079</pbbase:SampleMean>
        <pbbase:SampleMed>0.83167940378189087</pbbase:SampleMed>
        <pbbase:SampleStd>0.029663275550147809</pbbase:SampleStd>
        <pbbase:Sample95thPct>0.86801999807357788</pbbase:Sample95thPct>
        <pbbase:NumBins>30</pbbase:NumBins>
        <pbbase:BinCounts>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>12</pbbase:BinCount>
            <pbbase:BinCount>8</pbbase:BinCount>
            <pbbase:BinCount>11</pbbase:BinCount>
            <pbbase:BinCount>10</pbbase:BinCount>
            <pbbase:BinCount>17</pbbase:BinCount>
            <pbbase:BinCount>11</pbbase:BinCount>
            <pbbase:BinCount>20</pbbase:BinCount>
            <pbbase:BinCount>24</pbbase:BinCount>
            <pbbase:BinCount>24</pbbase:BinCount>
            <pbbase:BinCount>34</pbbase:BinCount>
            <pbbase:BinCount>25</pbbase:BinCount>
            <pbbase:BinCount>38</pbbase:BinCount>
            <pbbase:BinCount>52</pbbase:BinCount>
            <pbbase:BinCount>33</pbbase:BinCount>
            <pbbase:BinCount>45</pbbase:BinCount>
            <pbbase:BinCount>48</pbbase:BinCount>
            <pbbase:BinCount>57</pbbase:BinCount>
            <pbbase:BinCount>47</pbbase:BinCount>
            <pbbase:BinCount>69</pbbase:BinCount>
            <pbbase:BinCount>65</pbbase:BinCount>
            <pbbase:BinCount>55</pbbase:BinCount>
            <pbbase:BinCount>51</pbbase:BinCount>
            <pbbase:BinCount>57</pbbase:BinCount>
            <pbbase:BinCount>42</pbbase:BinCount>
            <pbbase:BinCount>29</pbbase:BinCount>
            <pbbase:BinCount>12</pbbase:BinCount>
            <pbbase:BinCount>3</pbbase:BinCount>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>1</pbbase:BinCount>
            <pbbase:BinCount>0</pbbase:BinCount>
            <pbbase:BinCount>1</pbbase:BinCount>
        </pbbase:BinCounts>

```

(continues on next page)

(continued from previous page)

```

    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
↪pbbase:MetricDescription>
  </pbds:ControlReadQualDist>
  <pbds:MedianInsertDist>
    <pbbase:SampleSize>973</pbbase:SampleSize>
    <pbbase:SampleMean>513.21990966796875</pbbase:SampleMean>
    <pbbase:SampleMed>350</pbbase:SampleMed>
    <pbbase:SampleStd>575.83020626506971</pbbase:SampleStd>
    <pbbase:Sample95thPct>1322</pbbase:Sample95thPct>
    <pbbase:NumBins>30</pbbase:NumBins>
    <pbbase:BinCounts>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>714</pbbase:BinCount>
      <pbbase:BinCount>101</pbbase:BinCount>
      <pbbase:BinCount>32</pbbase:BinCount>
      <pbbase:BinCount>18</pbbase:BinCount>
      <pbbase:BinCount>13</pbbase:BinCount>
      <pbbase:BinCount>8</pbbase:BinCount>
      <pbbase:BinCount>5</pbbase:BinCount>
      <pbbase:BinCount>6</pbbase:BinCount>
      <pbbase:BinCount>7</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>10</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>6</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>7</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>4</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>3</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>0</pbbase:BinCount>
      <pbbase:BinCount>2</pbbase:BinCount>
      <pbbase:BinCount>1</pbbase:BinCount>
      <pbbase:BinCount>9</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>93.333335876464844</pbbase:BinWidth>
    <pbbase:MinOutlierValue>289</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>289</pbbase:MinBinValue>
    <pbbase:MaxBinValue>2995.666259765625</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>6278</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Median Insert</pbbase:MetricDescription>
  </pbds:MedianInsertDist>
  <pbds:InsertReadLenDist>

```

(continues on next page)

```
<pbbase:SampleSize>901</pbbase:SampleSize>
<pbbase:SampleMean>4528.69384765625</pbbase:SampleMean>
<pbbase:SampleMed>5227</pbbase:SampleMed>
<pbbase:SampleStd>2322.8055598026981</pbbase:SampleStd>
<pbbase:Sample95thPct>7367</pbbase:Sample95thPct>
<pbbase:NumBins>30</pbbase:NumBins>
<pbbase:BinCounts>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>62</pbbase:BinCount>
  <pbbase:BinCount>39</pbbase:BinCount>
  <pbbase:BinCount>36</pbbase:BinCount>
  <pbbase:BinCount>29</pbbase:BinCount>
  <pbbase:BinCount>37</pbbase:BinCount>
  <pbbase:BinCount>19</pbbase:BinCount>
  <pbbase:BinCount>29</pbbase:BinCount>
  <pbbase:BinCount>37</pbbase:BinCount>
  <pbbase:BinCount>32</pbbase:BinCount>
  <pbbase:BinCount>32</pbbase:BinCount>
  <pbbase:BinCount>40</pbbase:BinCount>
  <pbbase:BinCount>45</pbbase:BinCount>
  <pbbase:BinCount>54</pbbase:BinCount>
  <pbbase:BinCount>73</pbbase:BinCount>
  <pbbase:BinCount>77</pbbase:BinCount>
  <pbbase:BinCount>97</pbbase:BinCount>
  <pbbase:BinCount>95</pbbase:BinCount>
  <pbbase:BinCount>49</pbbase:BinCount>
  <pbbase:BinCount>17</pbbase:BinCount>
  <pbbase:BinCount>2</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
  <pbbase:BinCount>0</pbbase:BinCount>
</pbbase:BinCounts>
<pbbase:BinWidth>418.89999389648438</pbbase:BinWidth>
<pbbase:MinOutlierValue>77</pbbase:MinOutlierValue>
<pbbase:MinBinValue>77</pbbase:MinBinValue>
<pbbase:MaxBinValue>12225.1025390625</pbbase:MaxBinValue>
<pbbase:MaxOutlierValue>12644</pbbase:MaxOutlierValue>
<pbbase:MetricDescription>Polymerase Read Length</
↪pbbase:MetricDescription>
</pbds:InsertReadLenDist>
<pbds:InsertReadQualDist>
  <pbbase:SampleSize>901</pbbase:SampleSize>
  <pbbase:SampleMean>0.82736450433731079</pbbase:SampleMean>
  <pbbase:SampleMed>0.83167940378189087</pbbase:SampleMed>
  <pbbase:SampleStd>0.029663275550147809</pbbase:SampleStd>
  <pbbase:Sample95thPct>0.86801999807357788</pbbase:Sample95thPct>
  <pbbase:NumBins>30</pbbase:NumBins>
  <pbbase:BinCounts>
    <pbbase:BinCount>0</pbbase:BinCount>
```

45

(continued from previous page)

```

        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>8</pbbase:BinCount>
        <pbbase:BinCount>11</pbbase:BinCount>
        <pbbase:BinCount>10</pbbase:BinCount>
        <pbbase:BinCount>17</pbbase:BinCount>
        <pbbase:BinCount>11</pbbase:BinCount>
        <pbbase:BinCount>20</pbbase:BinCount>
        <pbbase:BinCount>24</pbbase:BinCount>
        <pbbase:BinCount>24</pbbase:BinCount>
        <pbbase:BinCount>34</pbbase:BinCount>
        <pbbase:BinCount>25</pbbase:BinCount>
        <pbbase:BinCount>38</pbbase:BinCount>
        <pbbase:BinCount>52</pbbase:BinCount>
        <pbbase:BinCount>33</pbbase:BinCount>
        <pbbase:BinCount>45</pbbase:BinCount>
        <pbbase:BinCount>48</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>47</pbbase:BinCount>
        <pbbase:BinCount>69</pbbase:BinCount>
        <pbbase:BinCount>65</pbbase:BinCount>
        <pbbase:BinCount>55</pbbase:BinCount>
        <pbbase:BinCount>51</pbbase:BinCount>
        <pbbase:BinCount>57</pbbase:BinCount>
        <pbbase:BinCount>42</pbbase:BinCount>
        <pbbase:BinCount>29</pbbase:BinCount>
        <pbbase:BinCount>12</pbbase:BinCount>
        <pbbase:BinCount>3</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
        <pbbase:BinCount>0</pbbase:BinCount>
        <pbbase:BinCount>1</pbbase:BinCount>
    </pbbase:BinCounts>
    <pbbase:BinWidth>0.0049052196554839611</pbbase:BinWidth>
    <pbbase:MinOutlierValue>0.7500004768371582</pbbase:MinOutlierValue>
    <pbbase:MinBinValue>0.7500004768371582</pbbase:MinBinValue>
    <pbbase:MaxBinValue>0.89225196838378906</pbbase:MaxBinValue>
    <pbbase:MaxOutlierValue>0.89715707302093506</pbbase:MaxOutlierValue>
    <pbbase:MetricDescription>Polymerase Read Quality</
    <pbbase:MetricDescription>
    </pbds:InsertReadQualDist>
  </pbds:SummaryStats>
</pbds:DataSetMetadata>
</pbds:SubreadSet>

```

Example CcsreadSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:ConsensusReadSet
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="consensusreadset_150304_231155"
  MetaType="PacBio.DataSet.ConsensusReadSet"
  Name="DataSet_ConsensusReadSet"
  Tags="barcode moreTags mapping mytags"

```

(continues on next page)

(continued from previous page)

```

Version="2.3.0"
CreatedAt="2015-01-27T09:00:01">
<pbbase:ExternalResources>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5193"
    TimeStampedName="consensusread_bam_150304_231155"
    MetaType="PacBio.ConsensusReadFile.ConsensusReadBamFile"
    Name="First ConsensusRead BAM"
    Description="Points to an example ConsensusRead BAM file."
    ResourceId="file:///mnt/path/to/ccsreads0.bam"
    Tags="Example">
  <pbbase:FileIndices>
    <pbbase:FileIndex
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
      TimeStampedName="bam_index_150304_231155"
      MetaType="PacBio.Index.PacBioIndex"
      ResourceId="file:///mnt/path/to/ccsreads0.pbi"/>
    </pbbase:FileIndices>
  </pbbase:ExternalResource>
  <pbbase:ExternalResource
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
    TimeStampedName="consensusread_bam_150304_231155"
    MetaType="PacBio.ConsensusReadFile.ConsensusReadBamFile"
    Name="Second ConsensusRead BAM"
    Description="Points to an example ConsensusRead BAM file."
    ResourceId="file:///mnt/path/to/ccsreads1.bam"
    Tags="Example">
  <pbbase:FileIndices>
    <pbbase:FileIndex
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
      TimeStampedName="bam_index_150304_231155"
      MetaType="PacBio.Index.PacBioIndex"
      ResourceId="file:///mnt/path/to/ccsreads0.pbi"/>
    </pbbase:FileIndices>
  </pbbase:ExternalResource>
</pbbase:ExternalResources>
</pbds:ConsensusReadSet>

```

Example AlignmentSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:AlignmentSet
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  xmlns:pbbmeta="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatatypes.xsd"
  xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
↪pacificbiosciences.com/PacBioDataModel.xsd"
  Tags="barcode moreTags mapping mytags"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="alignmentset_150304_231155"
  MetaType="PacBio.DataSet.AlignmentSet"
  Name="Uber alignment set"
  Version="2.3.0"
  CreatedAt="2015-01-27T09:00:01">
  <pbbase:ExternalResources>

```

(continues on next page)

(continued from previous page)

```

<pbbase:ExternalResource
  Name="First Alignments BAM"
  Description="Points to an example Alignments BAM file."
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
  TimeStampedName="alignment_bam_150304_231155"
  MetaType="PacBio.AlignmentFile.AlignmentBamFile"
  ResourceId="file:///mnt/path/to/alignments0.bam" Tags="Example">
<pbbase:FileIndices>
  <pbbase:FileIndex
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
    TimeStampedName="bam_index_150304_231155"
    MetaType="PacBio.Index.PacBioIndex"
    ResourceId="file:///mnt/path/to/alignments0.pbi"/>
  </pbbase:FileIndices>
</pbbase:ExternalResources>
  <pbbase:ExternalResource
    Name="Reference used to produce BAM file"
    Description="Points to the reference used to generate the above_
↪BAM file."
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
    TimeStampedName="reference_fasta_150304_231155"
    MetaType="PacBio.ReferenceFile.ReferenceFastaFile"
    ResourceId="file:///mnt/path/to/reference.fasta" Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5593"
        TimeStampedName="sawriter_index_150304_231155"
        MetaType="PacBio.Index.SaWriterIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.sa"/>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5693"
        TimeStampedName="sam_index_150304_231155"
        MetaType="PacBio.Index.SamIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.fai"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
</pbbase:ExternalResource>
<!-- Is this second BAM necessary? <pbbase:ExternalResource Name="Second_
↪Alignments BAM" Description="Points to another example Alignments BAM file, by_
↪relative path." MetaType="PacBio.AlignmentFile.AlignmentBamFile" ResourceId="file://
↪alignments1.bam" Tags="Example">
  <pbbase:FileIndices>
    <pbbase:FileIndex MetaType="PacBio.Index.PacBioIndex" ResourceId=
↪"file:///mnt/path/to/alignments1.pbi"/>
  </pbbase:FileIndices>
</pbbase:ExternalResource>-->

</pbbase:ExternalResources>
<!-- START use IDREF here for External Resources, or just duplicate them?
↪<pbdsets:DataSets>
  <pbdsets:DataSet
    UniqueId="ab95d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="alignmentset_150304_231155"
    MetaType="PacBio.DataSet.AlignmentSet"
    Version="2.3.0"
    Name="HighQuality Read Alignments">

```

(continues on next page)

(continued from previous page)

```

    <pbbase:ExternalResources>
      <ExternalResource/>
    </pbbase:ExternalResources>
    <pbds:Filters> These Filters are in addition to those above. This
    →provides a means to subset and label the parent DataSet further.
      <pbds:Filter>
        <pbbase:Properties>
          <pbbase:Property Name="rq" Value="0.85" Operator=">"/>
        </pbbase:Properties>
      </pbds:Filter>
    </pbds:Filters>
  </pbds:DataSet>
  <pbds:DataSet
    UniqueId="ac95d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="alignmentset_150304_231155"
    MetaType="PacBio.DataSet.AlignmentSet"
    Version="2.3.0"
    Name="Alignments to chromosome 1">
    <pbbase:ExternalResources>
      <ExternalResource/>
    </pbbase:ExternalResources>
    <pbds:Filters>
      <pbds:Filter>
        <pbbase:Properties>
          <pbbase:Property Name="RNAME" Value="chr1" Operator="==" />
        </pbbase:Properties>
      </pbds:Filter>
    </pbds:Filters>
  </pbds:DataSet>
</pbds:DataSets>
-->
<pbds:DataSetMetadata>
  <pbds:TotalLength>50000</pbds:TotalLength>
  <pbds:NumRecords>5000</pbds:NumRecords>
  <pbds:Provenance CreatedBy="AnalysisJob">
    <pbds:ParentTool Name="pbalign" Version="0.1.0" Description="pbalign
    →subreads.dataset.xml reference.dataset.xml"/>
  </pbds:Provenance>
</pbds:DataSetMetadata>
</pbds:AlignmentSet>

```

Example ReferenceSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbds:ReferenceSet
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbds="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="referenceset_150304_231155"
  MetaType="PacBio.DataSet.ReferenceSet"
  Name="DataSet_ReferenceSet"
  Tags="barcode moreTags mapping mytags"
  Version="2.3.0"
  CreatedAt="2015-01-27T09:00:01">

```

(continues on next page)

(continued from previous page)

```

<pbbase:ExternalResources>
  <pbbase:ExternalResource
    Name="First References FASTA"
    Description="Points to an example references FASTA file."
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5293"
    TimeStampedName="referencefasta_150304_231155"
    MetaType="PacBio.ReferenceFile.ReferenceFastaFile"
    ResourceId="file:///mnt/path/to/reference.fasta"
    Tags="Example">
    <pbbase:FileIndices>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5393"
        TimeStampedName="sawriter_index_150304_231155"
        MetaType="PacBio.Index.SaWriterIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.sa"/>
      <pbbase:FileIndex
        UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe5493"
        TimeStampedName="sam_index_150304_231155"
        MetaType="PacBio.Index.SamIndex"
        ResourceId="file:///mnt/path/to/reference.fasta.fai"/>
      </pbbase:FileIndices>
    </pbbase:ExternalResource>
  </pbbase:ExternalResources>
<pbd:DataSetMetadata>
  <pbd:TotalLength>5000000</pbd:TotalLength>
  <pbd:NumRecords>500</pbd:NumRecords>
  <pbd:Organism>Tribble</pbd:Organism>
  <pbd:Ploidy>Diploid</pbd:Ploidy>
  <pbd:Contigs>
    <pbd:Contig
      Name="gi|229359445|emb|AM181176.4|"
      Description="Pseudomonas fluorescens SBW25 complete genome|quiver"
      Length="6722109"
      Digest="f627c795efad7ce0050ed42b942d408e"/>
    </pbd:Contigs>
  </pbd:DataSetMetadata>
</pbd:ReferenceSet>

```

Example ContigSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbd:ContigSet
  xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  xmlns:pbd="http://pacificbiosciences.com/PacBioDatasets.xsd"
  xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
  UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
  TimeStampedName="contigset_150304_231155"
  MetaType="PacBio.DataSet.ContigSet"
  Name="DataSet_ContigSet"
  Tags="HGAP"
  Version="2.3.0"
  CreatedAt="2015-01-27T09:00:01"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd">
  <pbbase:ExternalResources>
    <pbbase:ExternalResource
      UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe529c"

```

(continues on next page)

(continued from previous page)

```

        TimeStampedName="contig_fasta_150304_231155"
        MetaType="PacBio.ContigFile.ContigFastaFile"
        Name="First Contigs FASTA"
        Description="Points to an example contigs FASTA file e.g. from HGAP."
        ResourceId="file:///mnt/path/to/contigs.fasta" Tags="Example">
    </pbbase:ExternalResource>
</pbbase:ExternalResources>
<pbdns:DataSetMetadata>
    <pbdns:TotalLength>5000000</pbdns:TotalLength>
    <pbdns:NumRecords>500</pbdns:NumRecords>
    <pbdns:Contigs>
        <pbdns:Contig
            Name="gi|229359445|emb|AM181176.4|"
            Description="Pseudomonas fluorescens SBW25 complete genome|quiver"
            Length="6722109"
            Digest="f627c795efad7ce0050ed42b942d408e"/>
        </pbdns:Contigs>
    </pbdns:DataSetMetadata>
</pbdns:ContigSet>

```

Example BarcodeSet XML:

```

<?xml version="1.0" encoding="utf-8"?>
<pbdns:BarcodeSet
    xmlns:pbbase="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
    xmlns:pbdns="http://pacificbiosciences.com/PacBioDatasets.xsd"
    xmlns="http://pacificbiosciences.com/PacBioDataModel.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://pacificbiosciences.com/PacBioDataModel.xsd"
    UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe519c"
    TimeStampedName="barcodeset_150304_231155"
    MetaType="PacBio.DataSet.BarcodeSet"
    Name="DataSet_BarcodeSet"
    Tags="barcode moreTags mapping mytags"
    Version="2.3.0"
    CreatedAt="2015-01-27T09:00:01">
    <pbbase:ExternalResources>
        <pbbase:ExternalResource
            Name="First Barcodes FASTA"
            Description="Points to an example Barcodes FASTA file."
            UniqueId="b095d0a3-94b8-4918-b3af-a3f81bbe719c"
            MetaType="PacBio.BarcodeFile.BarcodeFastaFile"
            TimeStampedName="barcode_fasta_150304_231155"
            ResourceId="file:///mnt/path/to/barcode.fasta"
            Tags="Example"/>
        </pbbase:ExternalResources>
    <pbdns:DataSetMetadata>
        <pbdns:TotalLength>400</pbdns:TotalLength>
        <pbdns:NumRecords>30</pbdns:NumRecords>
        <pbdns:BarcodeConstruction>paired</pbdns:BarcodeConstruction>
    </pbdns:DataSetMetadata>
</pbdns:BarcodeSet>

```

5.8 Appendix 3: Use cases from pre-2.0 SMRT Analysis satisfied by the DataSet XML files

- **Refer to a set of subreads in multiple bax.h5 files**
 - FOFN of bax.h5 files (for blasr)
 - input.xml of bax.h5 files (for SMRT Pipe)
- **Refer to a subset of subreads by id from one or more bas.h5 files**
 - Whitelist option to P_Filter to generate a FOFN of rgn.h5 files + FOFN of bas.h5 files
- **Refer to a set of alignments from multiple different references or movies**
 - Explicitly merge the alignments into a larger (cmp.h5) alignment file
 - Create FOFN of cmp.h5 files
- **Run algorithms such as HGAP on a subset of subreads (e.g. that align to a contaminant such as E. coli)**
 - Awkward, but supported indirectly though whitelist option to P_Filter
- **Run algorithms such as Quiver on a subset of alignments (e.g. on a particular chromosome, or a particular chromosome region)**
 - Command line options to Quiver (to e.g. specify a particular reference). Not currently supported on other algorithms.
- **Refer to a subset of alignments that obey certain criteria. In particular, the extractBy reference or accuracy functionality**
 - Explicit creation of cmp.h5 files using cmph5tools.py select.
- **Perform any analysis that can be performed on an entire file of a particular data type (reads, read regions, alignments) on a particular reference**
 - Not supported pre-2.0.

CHAPTER 6

Run Design CSV specification for PacBio

The Run Design CSV is a comma-separated file which can be imported into SMRT Link to create a run design. Each line in the CSV represents a sample.

Key	Value Example	Value Example
Experiment Name	NoRS_Standard_Edna.1	Ca
Experiment Id	325/3250057	Mu
Experiment Description	20170530_A6_Iguana_VVnC_SampleSheet_TEMPLATE	Ca
Run Name	20170530_A6_Iguana_VVnC_SampleSheet_TEMPLATE	Ca
System Name	Sequel	Mu
Run Comments	ecoliK12_pbi_March2013	Ca
Is Collection	TRUE	Mu
Sample Well	A01	Mu
Sample Name	SMS_Iguana_A6_3230046_A01_TestCase_SB_BindKit_ChemKitv2_8rxnKit	Ca
Cell No.	1	Mu
Sequencing Mode	CLR	Mu
Generate CCS Data	FALSE	Mu
Movie Time per SMRT Cell (hours)	5	Mu
Use Predictive Loading	TRUE	Mu
Loading Target (P1 + P2)	0.4	Mu
Maximum Loading Time (hours)	1.2	Mu
Sample Comment	SMS_Iguana_A6_3230046_A01_TestCase_SB_BindKit_ChemKit	Ca
Insert Size (bp)	2000	Mu
On-Plate Loading Concentration (pM)	5	Mu
Size Selection	FALSE	Mu
Stage Start	FALSE	Mu
Reuse Sample	FALSE	Mu
Template Prep Kit Box Barcode	DM1117100259100111716	Mu
DNA Control Complex Box Barcode	DM1234101084300123120	Mu
Binding Kit Box Barcode	DM1117100862200111716	Mu
Sequencing Kit Box Barcode	DM0001100861800123120	Mu

Table 1 – continued from previous page

Key	Value Example	Value
Wash Kit Box Barcode	DM2222100866100123120	Mu
Automation Name	Diffusion	Ca
Automation Parameters	ExtensionTime=double:60 ExtendFirst=boolean:True	Mu
Primary Analysis	Default	Ca
Primary Analysis Parameters	CopyFileTrace=boolean:true	Mu
Sample is Barcoded	TRUE	Mu
Barcode Set	dad4949d-f637-0979-b5d1-9777eff62008	Mu
Same Barcodes on Both Ends of Sequence	TRUE	Mu
Barcode Name	lbc1-lbc1	Mu
Bio Sample Name	sample1	Ca
Pipeline Id	pbsmrtpipe.pipelines.sa3_ds_isoseq3_with_genome	Mu
Analysis Name	sample1 analysis	Ca
Entry Points	PacBio.DataSet.BarcodeSet;eid_barcode;afe89e3f-17ca-e9b8-eae9-b701dbb1f02d	A
Task Options	isocollapse.task_options.allow_extra_5exon;boolean>false	A

6.1 General Requirements

The csv may only contain ASCII characters. Specifically, it must satisfy the regular expression:

- `/^[\x00-\x7F]*$/g`

6.2 Required Fields

- Run Name
- Sample Well
- Sample Name
- Movie Time per SMRT Cell (hours)
- Insert Size (bp)
- Template Prep Kit Box Barcode
- Binding Kit Box Barcode
- Sequencing Kit Box Barcode

6.3 Is Collection

This field indicates whether the line is specifying a collection (TRUE), or a barcoded sample (FALSE). Collection lines should leave Barcode Names and Bio Sample Names blank. Barcoded sample lines only need to contain the Is Collection, Sample Name, the Barcode Name, and Bio Sample Name fields.

6.4 Experiment ID

Experiment IDs cannot contain the following characters: `<`, `>`, `:`, `"`, `\`, `|`, `?`, `*`, or `)`. Experiment IDs cannot start or end with a `/` and cannot have two adjacent `/`, i.e. `//`. Experiment IDs also cannot contain spaces.

6.9 Kit Barcodes

The kit barcodes are composed of three parts:

- Lot Number (ex: “DM1234”)
- Part Number (ex: “100-619-300”)
- Expiration Date (ex: “2020-12-31”)

which is used to make a single string. Using the above example, the barcode would be:

- DM1234100619300123120

Each kit must have a valid Part Number and cannot be obsolete. The list of kits can be found through a services endpoint such as:

- [server name]:[services port number]/smrt-link/bundles/chemistry-pb/active/files/definitions%2FPacBioAutomationConstraints.xml

This services endpoint will list, for each kit, the part numbers (“PartNumber”) and whether it is obsolete (“IsObsolete”). Dates must also be valid, meaning they must exist on the Gregorian calendar.

6.10 Parameters

The parameters are a “|” separated list. Each item follows the format: [parameter name]=[parameter type]:[parameter value]. Primary analysis parameters are:

- Readout
- MetricsVerbosity
- CopyFileTrace
- CopyFileBaz
- CopyFileDarkFrame
- CopyStatsH5

Acceptable parameter types are:

- String
- Int32
- UInt32
- Double
- Single
- Boolean
- DateTime

The parameter names and types are not case-sensitive.

6.11 Barcoded Sample Names

The barcoded sample names are a “|” separated list. Each item in the list follows the format: [barcode name];[biosample name] The barcode names must be contained within the specified barcodeset. A given barcode

name cannot appear more than once in the list. The biosample names can be any ASCII string but cannot contain the field separators “|” and “;”. The biosample names cannot be longer than 40 characters. A maximum of 384 barcodes is permitted per sample.

6.12 Auto Analysis fields

Auto Analysis is only supported on Sequel II. These fields include: Pipeline Id, Analysis Name, Entry Points, Task Options. You may define one analysis for each collection and bio sample. Pipeline Id, Analysis Name and Entry Points fields are required. The Task Options fields may be left empty, any task options not specified will use pipeline defaults.

CHAPTER 7

Legacy formats

7.1 PacBio legacy basecall File Format (bas.h5/bax.h5) Specification

The specification for bas.h5 and bax.h5 files can be found [here](#).

7.2 PacBio Alignment File Format (cmp.h5) Specification

7.2.1 Revision History

Version	Date	Authors	Comments
0.1	07/24/2009	Jason Chin	First draft
0.2	11/04/2009	Jason Chin	2nd draft, incorporated changes from prototype
0.3	11/17/2009	Susan Tang	Added consensus record
0.4	03/11/2009	Jason Chin, James Bullard	Added SF related spec and indexing proposal
0.5	07/06/2010	Dale Webster	Added PB Internal Format Spec
1.2rc	10/25/2010	Jason Chin, James Bullard, Dale Webster, Dimitris Iliopoulos, Ali Bashir	Major Revision before v. 1.2. Remove all reference to earlier Astro type cmp.h5. Meta-data group hierarchy changed. New attributes added. Define a few file operation behaviors. We call this document version 1.2rc to match the software release version for FCR. Preliminary support for strobe read timing information.
1.2	12/22/2010	Jason Chin	Finalize 1.2 spec , updated examples, revise the FileLog info group, remove TODO, remove “rc” in the version string.
1.3.1	03/6/2012	David Alexander, Mark Chaisson	QV record types changed. lastRow datasets removed. Converted to reStructuredText. Some material moved to Appendices
2.0.0	02/12/2013	David Alexander, James Bullard	Addition of chemistry tag information per-movie. Removal of master dataset constructs. Sortedness of a file now indicated by presence of OffsetTable.
2.1.0	08/01/2013	James Bullard	Addition of Barcode data
2.3.0	5/21/2014	David Alexander	Document revised chemistry encoding

7.2.2 File Format Versioning

The `cmp.h5` file format version is stored in the root group attribute `Version`. The version may take one of the following values:

- “1.2.0”
- “1.2.0.SF”
- “1.2.0.PB”
- “1.3.1.SF”
- “1.3.1.PB”
- “2.0.0”
- “2.1.0”
- “2.3.0”

File formats with versions ending in “.SF” (for Springfield) represent the production file formats that are produced by instruments at customer sites. File formats with versions ending in “.PB” (for PacBio) may contain additional information. *Version “X.PB” files are always usable wherever an “X.SF” file is usable; i.e. PacBio internal files contain a superset of the features required in a Springfield file, and the same formatting conventions are observed.*

7.2.3 Hierarchy Layout

In this section, we specify the general layout. At the top-level, or root group of the `cmp.h5` HDF5 file, there exist six HDF5 groups which must exist: `AlnInfo`, `RefInfo`, `MovieInfo`, `AlnGroup`, `RefGroup`, `FileLog`.

There are basically three different categories of data groups:

1. The *Info* groups contain information about particular aspects of the data contained in the file to some external references, e.g., reference sequences used for alignments, movies information for the reads, and ZMW hole numbers, etc. These groups will be referred to as info groups. (The only exception of such convention is the `FileLog` group. It should be considered as an Info group even though the group does have a “Info” suffix.)
2. The *Group* HDF5 groups contain information about how the data is stored in the file and function as key-value-pair mappings from integer IDs to character paths. Each “Group” HDF5 group will contain at least two datasets one of which will be called ID and the other will be called Path. The ID is the key used to refer to the HDF5 path stored in parallel in the Path dataset. To avoid ambiguity these groups will be referred to as mapping groups.
3. Additionally, at the top-level of the file, zero or more alignment data groups will exist—these groups contain the actual alignment data for each reference sequence and alignment group. These groups will be called data groups.

All datasets stored under the same HDF5 group irrespective of type shall always have the same number of rows or, in the case of dimensionless vectors, length.

Here we specify the minimal set of datasets in each of aforementioned groups:

1. An info group named `AlnInfo` containing information about each alignment stored in the file. The `AlnInfo` group should contain the following datasets:
 - a. `AlnIndex`: Dataset whose rows represent unique alignments and whose columns store relevant information about each alignment. The `AlnIndex` dataset has a string list attribute, `ColumnNames`, containing the names of the columns of this dataset.
 - b. (CCS only): A vector dataset `NumPasses`, of the same length as `AlnIndex`, indicating the number of CCS subreads that were used to generate the consensus read in the corresponding row of `AlnIndex`.
 - b. (Optional) Vector datasets, of the same length as `AlnIndex`, the same storing information about each alignment (e.g., `ZScore`, `SNR`, and `Edna`).
2. An info group named `RefInfo` containing information about the reference sequences used during alignment. The `RefInfo` group should contain the following datasets:
 - a. `ID`: Identifier of the record.
 - b. `FullName`: Name of the sequence as given by the FASTA file used during alignment.
 - c. `MD5`: md5 hashes of the DNA sequence used during alignment.

Note: The MD5 convention used in `cmp.h5` files differs from the standard convention in SAM files. SAM files store the “MD5 checksum of the sequence in the uppercase, with gaps and spaces removed.” *cmp.h5 files contain the MD5 checksums of the reference contig sequences as present in the reference FASTA file—case preserved, spaces and gaps intact (but newlines removed).*

- d. `Length`: The length of the DNA sequence used during alignment.
3. An info group named `MovieInfo` containing information about the movies which produced the alignments. This `MovieInfo` group should contain the following datasets:
 - a. `ID`: Identifier of the record.

- b. Name: Movie name.
- c. **FrameRate:** The camera speed in frames per second used to record the movie.
- d. Datasets encoding information about the sequencing chemistry that was used. This is encoded in one of two manners:
 - 1. Datasets `SequencingKit`, `BindingKit`, and `SoftwareVersion` represent the partnumbers read by the instrument barcode reader for each movie run, as well as the basecaller version. Decoding of this identifying “triple” for each movie is deferred to the tools that actually need to know the chemistry details—specifically, the Quiver variant/consensus calling tool and the base-modification identification tools.
 - 2. (*Versions 2.2.0 and earlier, and manual override in 2.3.0 and after*) Dataset `SequencingChemistry`, representing a canonical string representation (for example, “P4-C2”) of the chemistry. Note that this places the burden for decoding of the barcode information on the software that constructs the `cmp.h5` rather than client software.

Software that parses the `cmp.h5` format shall rely on the datasets in (1) as the canonical chemistry information, only falling back to the information in (2) if the datasets in (1) are absent.
- 4. An info group named `FileLog` containing information about the history of the file itself.
 - a. ID: Identifier of the record
 - b. Program: The name of the program that touches the file
 - c. Version: The version of the program that touches the file
 - d. Timestamp: A **W3C compatible timestamp** string of the date-time when the file is touched.
 - e. CommandLine: Detail command line string that details how the program is used
 - f. Log: The field to store any extra details
- 5. A mapping group named `RefGroup` that records the reference sequence information used in the alignments: The `RefGroup` group should contain the following datasets:
 - a. ID
 - b. Path
 - c. `RefInfoID`: `RefInfoID` refers to elements of the `/RefInfo/ID` dataset.
- 6. A mapping group named `AlnGroup` that records the different partitions of alignments. This data group should contain:
 - a. ID
 - b. Path
- 7. Zero or more data groups containing the actual alignments. The names of the groups are defined by the dataset `/RefGroup/Path`. Each reference group contains one or more alignment groups (representing alignments from some predefined grouping, such as: `SMRTcell`, acquisition, or movie, etc). The full HDF5 paths to the alignment groups including the group names are defined in the dataset `/AlnGroup/Path`. An alignment group should contain:
 - a. A single alignment array dataset named `AlnArray`
 - b. (Optional) Datasets for quality values and pulse features that can be aligned to the read bases. Detailed information about necessary datasets is defined in sections 10 and 11.

8. (Optional) User-defined datasets conforming to the conventions of simple HDF5 types and having the same length as each sibling in its containing group.

It may be helpful to inspect the output of *h5ls* applied to a 1.3.1.SF cmp.h5 file:

```
mp-f052:~ $ h5ls -r ~/Data/new_cmph5/alignments.cmp.h5
/
/AlnGroup          Group
/AlnGroup/ID       Dataset {1/Inf}
/AlnGroup/Path     Dataset {1/Inf}
/AlnInfo           Group
/AlnInfo/AlnIndex  Dataset {16866/Inf, 22/Inf}
/FileLog           Group
/FileLog/CommandLine Dataset {3/Inf}
/FileLog/ID        Dataset {3/Inf}
/FileLog/Log       Dataset {3/Inf}
/FileLog/Program   Dataset {3/Inf}
/FileLog/Timestamp Dataset {3/Inf}
/FileLog/Version   Dataset {3/Inf}
/MovieInfo         Group
/MovieInfo/FrameRate Dataset {1/Inf}
/MovieInfo/SequencingChemistry Dataset {1/Inf}
/MovieInfo/ID      Dataset {1/Inf}
/MovieInfo/Name    Dataset {1/Inf}
/RefGroup          Group
/RefGroup/ID       Dataset {1/Inf}
/RefGroup/OffsetTable Dataset {1/Inf, 3/Inf}
/RefGroup/Path     Dataset {1/Inf}
/RefGroup/RefInfoID Dataset {1/Inf}
/RefInfo           Group
/RefInfo/FullName  Dataset {1/Inf}
/RefInfo/ID        Dataset {1/Inf}
/RefInfo/Length    Dataset {1/Inf}
/RefInfo/MD5       Dataset {1/Inf}
/ref000001         Group
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0 Group
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/AlnArray_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/DeletionQV_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪DeletionTag Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/IPD_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪InsertionQV Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/MergeQV_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/PulseWidth_
↪Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪QualityValue Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪SubstitutionQV Dataset {39434696/Inf}
/ref000001/m120225_045819_richard_c100304312550000001523012308061200_s1_p0/
↪SubstitutionTag Dataset {39434696/Inf}
```


7.2.4 Root Group Attributes

The following mandatory string attributes should be set in the root group:

Name	Allowed Values	Comment
Version	"1.2.0" "1.2.0.SF" "1.2.0.PB" "1.3.1.SF" "1.3.1.PB" "2.0.0"	The suffix is used to indicate whether the file includes (".SF") or does not include (".PB") several datasets useful for in-house analyses.
Read-Type	"RCCS", "CCS", "strobe", "standard", or "cDNA"	Set to "standard" by default. If the cmp.h5 is used for "RCCS" and "CCS", there will be no pulse features. Each read type will allow different sets of optional tables.
Command-Line	The command line used for generating this file.	This attribute is reserved for the initial generation. All post-initial alignment information should be stored in FileLog

7.2.5 Mapping Groups: ID, and Path datasets

Each mapping group contains at least an `ID` and `Path` dataset. The `ID` dataset contains unique positive integer values. The `Path` dataset contains proper HDF5 paths to HDF5 groups within the file. Elements of the path dataset should conform to the following regular expression (leading forward slash not included):

"[a-zA-Z+_0-9]+" (all lower and upper case ASCII characters, numbers, "-", and "+").

The `ID`, `Path` datasets function as key-value pair mappings. The individual IDs are used in datasets to reference the relevant information stored in this particular mapping group.

The following **HDF5 DDL** defines the hdf5 data types for these data sets:

```

DATASET "ID" {
    DATATYPE  H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}
DATASET "Path" {
    DATATYPE  H5T_STRING {
        STRSIZE H5T_VARIABLE;
        STRPAD  H5T_STR_NULLTERM;
        CSET    H5T_CSET_ASCII;
        CTYPE   H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}

```

Two datasets are used to avoid compound types in an HDF5 file. This avoids the complication in reader/writer code implementations. If there is a mature compound type code base within the PBI development environment, compound type datasets are recommended for storing such key-value pairs.

7.2.6 RefGroup data group and /RefGroup/* datasets

The `RefGroup` mapping group provides a mapping between reference sequence identifiers (`ID`) to HDF5 paths in the file (`Path`). An example HDF5 schema can be seen above. A `RefInfoID` data set is used for pointing to the `ID` dataset in the `RefInfo` group and can be viewed as a foreign key.

The following DDL code block defines the data types for the datasets and attributes associated with `/RefGroup`:

```

GROUP "RefGroup" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Path" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "RefInfoID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.7 AlnGroup data group: /AlnGroup/* datasets

The AlnGroup mapping group provides a mapping between alignment group identifiers (ID) to alignment group paths.

The following DDL code block defines the data types for the datasets and attributes associated with /AlnGroup:

```

GROUP "AlnGroup" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Path" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.8 RefInfo info group and /RefInfo/* datasets

The RefInfo info group provides information about the reference sequences used during alignment. The RefInfo group contains at least 4 datasets including the ID dataset. The RefInfo/FullName provides the name of the sequence aligned to and is the full FASTA name. The RefInfo/MD5 is an MD5 hash of the reference sequence aligned to. The RefInfo/Length provides the length of the sequence aligned to.

Other sequence specific annotations can be stored as parallel datasets at this level.

The following DDL code block defines the data types for the datasets and attributes associated /RefInfo:

```

GROUP "RefInfo" {
  DATASET "FullName" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Length" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "MD5" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.9 MovieInfo data group: MovieInfo/* datasets

The paired arrays MovieInfo/ID and MovieInfo/Name in the MovieInfo group are defined to indicate the source of the movies for the reads in the AlnInfo/AlnIndex dataset. This pair of arrays functions as a key-value-pair map between IDs and movie names.

The following DDL code block defines the data types for the datasets and attributes associated /MovieInfo:

```

GROUP "MovieInfo" {
  DATASET "ID" {
    DATATYPE H5T_STD_U32LE
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
  DATASET "Name" {
    DATATYPE H5T_STRING {
      STRSIZE H5T_VARIABLE;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
    DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
  }
}

```

7.2.10 AlnInfo data group and the AlnArray data sets

AlnInfo data group

The first column of the `AlnIndex` can be treated as the equivalent “ID” dataset in the mapping or the info groups.

The data types of the dataset `AlnIndex` are defined as:

```

DATASET "AlnIndex" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( *, 22 ) / ( H5S_UNLIMITED, 22 ) }
}

```

AlnIndex dataset

The purpose of the `AlnIndex` dataset is to:

1. Store the information necessary to retrieve alignments from the file. This includes: path, beginning offset, and ending offset within the dataset containing the alignment. (This kind of reference to alignment is similar to that proposed by HDF5 group in the bioHDF5 specification.)
2. Store the information, e.g., the orientation (i.e., strand) of the alignment, for processing the alignment properly for downstream bioinformatics analysis and visualization.
3. Store information that can be used to indentify the original reads.
4. Store the unique unsigned 32 bit integer ID as single unique key for each individual alignment.
5. Store summary information about the alignment. For example, one can store the number of matches, mismatches, insertions, deletions, mapping quality, read level quality values, etc.

AlnIndex Dataset Columns

The 22 columns in the `AlnIndex` dataset are described in the table below.

PacBioFileFormats Documentation, Release 11.0.0

Column Name	Meaning	Comment
AlnID	Non-zero unique 32 bit integer key for the alignment record	Each alignment should have a unique AlnID. No other assumption about the order of the AlnID should be used for data processing.
AlnGroupID	A foreign key referring to AlnGroup/ID	
MovieID	A foreign key referring to MovieInfo/ID	
RefGroupID	A foreign key referring to RefGroup/ID.	
tStart	The start position (0-based, inclusive) of the alignment target (the reference sequence)	tStart should always be less than tEnd, even when the hit is against the opposite strand.
tEnd	The end position (0-based, not-inclusive) of the alignment target (the reference sequence)	tEnd should always be greater than tStart, even when the hit is against the opposite strand.
RCRefStrand	The relative strand in the alignment. 1 for reversed reference strand; 0 for forward-forward alignment	The read base should never be reverse-complimented in the alignment array, so we only need to record if the reference bases are presented in reverse complemented strand in the file. "1" means "Yes/True" here.
HoleNumber	The HoleNumber from the bas.h5	
SetNumber		
StrobeNumber ExonNumber	Context dependent value. When the read type is Strobe, this field is the strobe number. When the read type is cDNA it will be the exon number.	
MoleculeID	An integer which is unique to all subreads from the same ZMW.	If multiple subreads are from the same physical origin, they should have the same MoleculeID and different physical origins should have different MoleculeID.
rStart	The start position (0-based, inclusive) of the read in the alignment	Regardless weather the alignment is a subread or not, the position is always relative to the original raw full read sequence.
rEnd	The end position (0-based, not-inclusive) of the read in the alignment	rEnd should always be greater than rStart.
MapQV	TBD	
nM	Number of matched base in the alignment	
nMM	Number of mis-matched base in the alignment	
nIns	Number of insertions in the read relative to the reference sequence	
nDel	Number of deletions (missing bases) in the read relative to the reference sequence	
Offset_begin	The beginning position (0-based, inclusive) of the alignment in the AlignmentArray	
Offset_end	The ending position (0-based, exclusive) the alignment in the AlignmentArray	Not including the padded zero of the alignment array.
nBackRead	Used for faster access to blocks of sorted reads	See the sorting and indexing section
nReadOverlap	Used for faster access to blocks of sorted reads	See the sorting and indexing section

The column names should be stored as an attribute `ColumnNames` that contains all names listed in "Column Name" in the table above.

7.2.11 Sequence Alignments

Binary Encoding for Alignment Pair

The *alignment array* is a one dimensional 8 bit unsigned integer array where the individual array elements represent a “read base - reference base” pair packed into one byte. The higher four bits are set by the read base and the lower four bits are set by the reference base as the following:

```
0 0 0 0 0 0 0 0
T G C A T G C A
```

For example, “T” and “T” matched alignment will be presented as 0b10001000=136. “T” vs. “G” mismatch will be represented as 0b10000100=132. Insertion of “T” in read will be 0b10000000=128. “No-call” (“N”) bases are encoded as 0b1111=15 for both read and reference.

In the *AlnArray* dataset, the encoded read base should be always the same as what has been observed by the sequencing machine without any complementation. If a read is aligned to the reverse complement strand of the reference sequence, the lower four bits represent the complemented base (i.e., the reference has been complemented).

Alignment Array

The example below shows the conversion of an alignment pair to the binary array represented as an integer:

```
Alignment:

  Read Bases: ATCTT--ATC-GTTAATTA--A
  Ref. Bases: A-CTCAGA-CAGTCAATTAGCA

Encoded Alignment Pairs:

AA -> 17
T- -> 128
CC -> 34
TT -> 136
TC -> 130
-A -> 1
-G -> 4
...
-C -> 2
AA -> 17
```

The final encoded array for this alignment is [17, 128, 34, 136, 130, 1, 4, 17, 128, 34, 1, 68, 136, 130, 17, 17, 136, 136, 17, 4, 2, 17, 0].

Note that zero is padded at the end of each alignment as a separator between different alignments. This will enable some analysis by simply streaming the alignment array without extra index look-ups to separate different alignments.

The alignment array is a concatenation of all encoded alignment arrays of each read and the *AlignmentIndex* dataset is used to identify the origin of each alignment.

Below is an example of the HDF5 type definition for an *AlnArray*:

```
DATASET "AlnArray" {
  DATATYPE H5T_STD_U8LE
  DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) }
}
```

7.2.12 Pulse Metrics and QVs

In addition to the basic and required `AlnArray` dataset present in each alignment group, pulse metrics and quality values (QVs) may be optionally provided; however if one of these features is provided for one alignment group they must be provided for all alignment groups. These optional datasets are:

- `DeletionQV`,
- `DeletionTag`,
- `InsertionQV`,
- `MergeQV`,
- `SubstitutionQV`,
- `SubstitutionTag`,
- `QualityValue`,
- `IPD`,
- `PulseWidth`,
- `StartFrame`,
- `pkmid`

Each such dataset is of the same shape as the `AlnArray` dataset in the same alignment group. Missing values (corresponding to read gaps in the alignment array) are encoded based on the type of the dataset:

Data type	Missing value encoding
<code>float32</code>	NaN
<code>int8 (char)</code>	'-' (ASCII 42)
<code>uint8</code>	255
<code>uint16</code>	65535

A missing value is present at a dataset offset if and only if that offset corresponds to a read gap in the `AlnArray`.

For the types of the pulse metric and QV datasets, see [Summary of Attributes and Datasets](#). Any offset into a pulse metric or QV dataset corresponds to the same offset in the `AlnArray`.

7.2.13 Specification for the `cmp.h5` used for automatic data analysis from the instrument

This section defines the constraints that a `cmp.h5` file should satisfy for automatic data analysis for an Springfield instrument. Such files are labeled with a root group attribute `Version` of “1.2.0.SF” or “1.3.1.SF”.

The `RefGroup/Path` for 1.2.0.SF and 1.2.0.PB `cmp.h5` files has the form of “ref%06d” (C string formatting convention). The original FASTA sequence header should be stored in the `RefInfo/FullName` dataset. Additionally, two other datasets are obligatory: `RefInfo/Length` and `RefInfo/MD5`.

The default of `AlnGroup` partition is to group alignments from the same movie that aligned to the same reference together and we use the movie filename without suffix as the default alignment group name.

7.2.14 Specification for the `cmp.h5` used for PacBio internal data analysis

In addition to all datasets specified for the standard `cmp.h5` the following additional datasets are required in internal files (“1.2.1.PB”):

1. Within the info group named “MovieInfo” containing information about the movies which produced the alignments:
 - Exp: A uint32 dataset specifying the PacBio LIMS Experiment code associated with each movie in the corresponding /MovieInfo/Name dataset.
 - Run: A uint32 dataset specifying the PacBio LIMS Run code associated with each movie in the corresponding /MovieInfo/Name dataset.

Data type and data space definition:

```

DATASET "/MovieInfo/Exp" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( 1 ) / ( H5S_UNLIMITED ) }
}
DATASET "/MovieInfo/Run" {
  DATATYPE  H5T_STD_U32LE
  DATASPACE SIMPLE { ( 1 ) / ( H5S_UNLIMITED ) }
}

```

2. Within the info group named AlnInfo containing information about each alignment stored in the file:
 - ZScore: a float32 dataset containing the alignment significance score (“Z Score”) computed from the corresponding row of the /AlnInfo/Index table.

Data type and data space definition:

```

DATASET "/AlnInfo/ZScore" {
  DATATYPE  H5T_IEEE_F32LE
  DATASPACE SIMPLE { ( 310 ) / ( H5S_UNLIMITED ) }
}

```

3. In addition to all attributes specified for the standard cmp.h5 the following additional root level attributes are required:

Attribute name	Type	Sample values	Comment
ReportsFolder	string	“Analysis_Reports”	Contains the directory name of the Primary Analysis Reports used for this alignment
PrimaryPipeline	string	“61453”	Contains the Perforce changelist number of the Primary Analysis Pipeline used for this alignment

7.2.15 Sorting, Flattening, Merging, Splitting and Filtering Behaviors

Sorting

In order to provide fast access to cmp.h5 files, we provide sorted cmp.h5 files. These files have some additional information to quickly retrieve contiguous regions according to an indexing scheme. The most typical use case is to obtain a set of reads overlapping a particular genomic region, where the region can be a single genomic coordinate or ranges of genomic coordinates. Note that by default, *sorting* only entails the sorting of the AlnIndex dataset, and not the sorting of the alignment data itself.

A sorted cmp.h5 file has the following additional items as compared to an unsorted cmp.h5:

1. A dataset OffsetTable stored within the RefGroup mapping group giving the offsets of the reads mapped to a reference sequences in the global alignment index. The dataset is a 3 by N unsigned 32 bit unsigned integer array, where N is the total number of reference sequences in the RegGroup/ID table. The three elements of each row in the array indicate the RefID, targetStartOffset, and targetEndOffset.

The `targetStartOffset` and `targetEndOffset` give the range of the reads in the global `/AlnInfo/AlnIndex` that maps to the specific reference sequence in the first column of the dataset. The presence or absence of the `OffsetTable` dataset should be used to determine whether the file is sorted or unsorted./

2. The alignment index will have two additional columns of unsigned 32-bit integers (these could be shorter) `nBackRead` and `nReadOverlap` which gives the maximum number of reads one needs to examine to determine overlap and the actual number of reads which overlap a position, respectively. A value of -1 indicates that the field has not been filled in, whereas a value of 0 means that no further reads possibly overlap the position of interest. Here, `nBackRead > nReadOverlap` is always true.
3. In addition to sorting the `AlnIndex`, sorting and indexing can perform a “flattening” operation whereby all `AlnGroups` under each `RefGroup` are merged into a single `AlnGroup`. The name of the single `AlnGroup` can be anything, however, convention is to use the name: “rg-0001” to indicate that the sub-datasets have been merged and re-ordered. Additionally, an attribute on this group: `repacked` will be set to 1 to indicate, irrespective of the name, that the datasets have been sorted. If the length of any of the child datasets of a “repacked” alignment group would be greater than 2^{32} , then additional alignment groups are added serially, e.g., “rg-0002”, etc. An alignment will never span more than one alignment group.

Note: The time complexity of sorting a `cmp.h5` file will be on the order of $O(n \log(n))$. Additionally, the columns `nBackRead` and `nReadOverlap` need to be computed. This will be on the order of $O(\max(\text{read length}) * n)$. Access to a given start position in `cmp.h5` will be $O(\log(n))$, however, this will only produce reads having that start position. In order to obtain all reads overlapping a position, one needs to inspect the `nBackRead` to obtain the size of the slice that they should grab from the `cmp.h5` file. Retrieval, therefore, is bound by $O(n \text{BackRead} \log(n))$. The additional column, `nReadOverlap`, should allow one to obtain significantly better performance, as the search can stop once to obtained number of reads is equal to `nReadOverlap`.

Merging

Merging is performed on a list of `cmp.h5` files by selecting the first file to act as the seed and sequentially merging the rest onto the seed. If the first file in the list of files to be merged is empty then the next non-empty file is selected to act as the seed. An exact copy of the seed is made where all ID-type datasets have their entries serialized to consecutive 32 bit integers starting from 1. Merging results modifies a copy of the seed file in place. For each `cmp.h5` file in the merging list, the following steps are performed:

1. If the file is empty, its root group `Version` does not match the seed’s `Version` or does not have the same type of loaded `PulseMetrics` as the seed, it is removed from the merging list and the next file is considered.
2. Root group attributes are not merged since they are set to the seed’s Root group attributes.
3. Datasets under the seed’s `AlnInfo` Data group are extended with their counterparts from the file to be merged. The `AlnID` column of the newly added rows in the `AlnInfo/Index` is updated by resetting the old values from the merged file. The new values are set equal to a list of integers starting from the maximum `AlnID` of the seed + 1, adding 1 for each new `AlnID` from the merged file.
4. Datasets under the seed’s `RefInfo`, `MovieInfo`, `AlnGroup` and `RefGroup` data groups are extended only with new entries from their counterparts in the file to be merged. If new `RefInfo/ID`, `RefGroup/ID` or `MovieInfo/ID` entries are created, they are mapped back to their respective columns in the `AlnInfo/Index`.

After going through the entire list of files to be merged, the `FileLog` attribute from the Root group attributes is modified (TBD).

Splitting

The current splitting behavior is implementation specific and associated with a single use case, i.e., processing of `.cmp.h5` files involved in Edna analysis- type workflows. It is our aim to generalize the splitting behavior to accom-

moderate more use cases when those become available.

A master cmp.h5 file is split into an N number of cmp.h5 files where N is equal to the number of RefInfo/ID entries in the master file. Consequently, each new cmp.h5 file contains all data associated with a single reference sequence. This is done by:

1. Creating N copies of the master cmp.h5 file and sequentially selecting a RefInfo/ID entry to become the only entry for each copied file, unique amongst the group.
2. Resizing all datasets belonging to AlnInfo, RefInfo, MovieInfo, AlnGroup and RefGroup by deleting all entries that are not associated with the chosen reference sequence. Splitting maintains the values of all ID-type fields and data fields in the AlnInfo/Index rows.
3. Maintaining the size and content of the AlnArray and PulseMetric-type datasets in the new files as the ones in the master.

7.2.16 Barcode Information

In addition to the aforementioned core alignment information, the cmp.h5 file can be used to store optional datasets containing barcode annotation on alignments. The pattern leveraged to store this annotation demonstrates a general mechanism to extend the information stored in the cmp.h5 file for downstream applications.

In the case of barcoding, we wish to label alignments according to their barcode so that other applications can leverage this information when computing statistics over sets of alignments, e.g., consensus calling within sample. To this end, a parallel dataset to /AlnInfo/AlnIndex is created. The Barcode dataset is 32-bit integer matrix with the same number of rows as the AlnIndex dataset and 5 columns storing scoring and labeling information.

The Barcode dataset contains the total number of barcodes scored for this molecule (count), the index of the top-scoring barcode (index1), the score of the top-scoring barcode (score1), the index of the 2nd-highest scoring barcode (index2) and its score (score2). These columns are named in the attribute ColumnNames of the Barcode dataset.

The index1 and index2 are foreign-keys into the BarcodeInfo/ID dataset. Analogous to the other *Info datasets, the BarcodeInfo/ID and BarcodeInfo/Name are used to retrieve the human-readable name of the barcode.

7.2.17 Summary of Attributes and Datasets

Versions prior to 2.0.0 are described in the Appendices.

File Version 2.0.0 contents:

Parent Group	HDF5 data	Resource Name	Data type	Shape	
/	ATTR	CommandLine	VLEN_STR	None	required
/	ATTR	Index	VLEN_STR	(3,)	optional
/	ATTR	ReadType	VLEN_STR	None	required
/	ATTR	Version	VLEN_STR	None	required
/AlnGroup	DS	ID	uint32	1	required
/AlnGroup	DS	Path	VLEN_STR	1	required
/AlnInfo	DS	AlnIndex	uint32	22	required
/AlnInfo	ATTR	ColumnNames	VLEN_STR	22	required
/FileLog	DS	CommandLine	VLEN_STR	1	required
/FileLog	DS	ID	uint32	1	required
/FileLog	DS	Log	VLEN_STR	1	required

Continued on next page

Table 1 – continued from previous page

Parent Group	HDF5 data	Resource Name	Data type	Shape	
/FileLog	DS	Program	VLEN_STR	1	required
/FileLog	DS	Timestamp	VLEN_STR	1	required
/FileLog	DS	Version	VLEN_STR	1	required
/MovieInfo	DS	ID	uint32	1	required
/MovieInfo	DS	Name	VLEN_STR	1	required
/MovieInfo	DS	FrameRate	float32	1	required
/MovieInfo	DS	SequencingChemistry	VLEN_STR	1	required
/ref*/*	DS	AlnArray	uint8	1	required
/ref*/*	DS	QualityValue	uint8	1	optional
/ref*/*	DS	DeletionQV	uint8	1	optional
/ref*/*	DS	InsertionQV	uint8	1	optional
/ref*/*	DS	MergeQV	uint8	1	optional
/ref*/*	DS	SubstitutionQV	uint8	1	optional
/ref*/*	DS	SubstitutionTag	char	1	optional
/ref*/*	DS	DeletionTag	char	1	optional
/ref*/*	DS	IPD	uint16	1	optional
/ref*/*	DS	PulseWidth	uint16	1	optional
/ref*/*	DS	PulseIndex	uint32	1	optional
/RefGroup	DS	ID	uint32	1	required
/RefGroup	DS	OffsetTable	uint32	3	optional
/RefGroup	DS	Path	VLEN_STR	1	required
/RefGroup	DS	RefInfoID	uint32	1	required
/RefInfo	DS	FullName	VLEN_STR	1	required
/RefInfo	DS	ID	uint32	1	required
/RefInfo	DS	Length	uint32	1	required
/RefInfo	DS	MD5	VLEN_STR	1	required
/BarcodeInfo	DS	ID	uint32	1	optional
/BarcodeInfo	DS	ID	uint32	1	required
/BarcodeInfo	DS	Name	VLEN_STR	1	required

CHAPTER 8

APIs available

We occasionally make changes to these file format specifications so we recommend using PacBio-authored APIs to access these file types.

- C++: [pbbam](#)
- Python: [pbcore](#)

CHAPTER 9

Data Model XSD

For completeness, here is the PacBio data model XSD.

9.1 PacBioDataModel

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://
pacificbiosciences.com/PacBioDataModel.xsd" xmlns:pbbase="http://pacificbiosciences.
com/PacBioBaseDataModel.xsd" xmlns:pbpn="http://pacificbiosciences.com/
PacBioPartNumbers.xsd" xmlns:pbdmeta="http://pacificbiosciences.com/
PacBioCollectionMetadata.xsd" xmlns:pbdbs="http://pacificbiosciences.com/
PacBioDatasets.xsd" xmlns:pbsample="http://pacificbiosciences.com/PacBioSampleInfo.
xsd" targetNamespace="http://pacificbiosciences.com/PacBioDataModel.xsd"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioCollectionMetadata.
xsd" schemaLocation="PacBioCollectionMetadata.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioDatasets.xsd"
schemaLocation="PacBioDatasets.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioPartNumbers.xsd"
schemaLocation="PacBioPartNumbers.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
schemaLocation="PacBioSampleInfo.xsd"/>
  <xs:element name="Assay" type="AssayType"/>
  <xs:element name="ChipLayout">
    <xs:annotation>
      <xs:documentation>Part of the RunResources; specifies a
ChipLayout which is compatible with the collection protocols defined on the plate</
xs:documentation>
```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Validation" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="Name" type="xs:string" use="required"/>
            <xs:attribute name="PartNumber" type="xs:string" use="required
↪"/>

            <xs:attribute name="Quantity">
                <xs:annotation>
                    <xs:documentation>The number of cells_
↪required, of a particular part number</xs:documentation>
                </xs:annotation>
            </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="CompatibleChipLayouts">
        <xs:annotation>
            <xs:documentation>A set of Chip Layouts deemed compatible_
↪with the current plate</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="ChipLayout" maxOccurs="unbounded"/>
                <xs:element ref="Validation" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="CompatibleSequencingKits">
        <xs:annotation>
            <xs:documentation>A set of reagent kits deemed compatible_
↪with the current plate</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="pbpn:SequencingKit"/>
                <xs:element ref="RequiredTips"/>
                <xs:element ref="EstimatedTotalRunTime"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="EstimatedTotalRunTime">
        <xs:annotation>
            <xs:documentation>The total amount of time the run is_
↪estimated to require. A confidence value (defaulted to 90%) indicates the degree_
↪of certainty associated with the estimate</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Validation" minOccurs="0"/>
            </xs:sequence>
            <xs:attribute name="Value" type="xs:string" use="required"/>
            <xs:attribute name="Confidence" type="xs:int" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="Events" type="pbbase:RecordedEventType"/>
    <xs:element name="Input" type="pbbase:InputOutputDataType"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Output" type="pbbase:InputOutputDataType"/>
<xs:element name="Parameter">
  <xs:annotation>
    <xs:documentation>A variable, as a name/value pair,
↪ associated with a protocol (one of Collection, Primary, and Secondary)</
↪ xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Validation" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Name" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value=
↪ "AverageReadLength"/>
          <xs:enumeration value="AcquisitionTime
↪ "/>
          <xs:enumeration value="InsertSize"/>
          <xs:enumeration value="ReuseComplex"/>
          <xs:enumeration value="StageHS"/>
          <xs:enumeration value="JobId"/>
          <xs:enumeration value="JobName"/>
          <xs:enumeration value=
↪ "NumberOfCollections"/>
          <xs:enumeration value="StrobeByTime"/>
          <xs:enumeration value="UsedControl"/>
          <xs:enumeration value="Use2ndLook"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="Value" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>
<xs:element name="PacBioDataModel">
  <xs:annotation>
    <xs:documentation>PacBio Data Model root element</
↪ xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ExperimentContainer" type=
↪ "ExperimentContainerType"/>
      <xs:any minOccurs="0">
        <xs:annotation>
          <xs:documentation>By using the "any"
↪ element we can extend (after "ProjectContainer") the content of "PacBioDataModel"
↪ with any element</xs:documentation>
        </xs:annotation>
      </xs:any>
    </xs:sequence>
    <xs:attribute name="Version" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation>An optional identifier
↪ denoting the revision of this particular entity</xs:documentation>
      </xs:annotation>
    </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        </xs:complexType>
      </xs:element>
      <xs:element name="RequiredSMRTCells">
        <xs:annotation>
          <xs:documentation>Part of the RunResources; specifies the_
↪required number of SMRT cells</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Quantity" type="xs:int" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="RequiredTips">
        <xs:annotation>
          <xs:documentation>Part of the RunResources; specifies the_
↪required number of tips via two attributes, Left and Right</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Left" type="xs:int" use="required"/>
          <xs:attribute name="Right" type="xs:int" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="RunResources">
        <xs:annotation>
          <xs:documentation>This is an output field specifying the_
↪requirements for the run, e.g. number of tips, estimated run time, etc.</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="CompatibleSequencingKits" maxOccurs=
↪"unbounded"/>
            <xs:element ref="CompatibleChipLayouts"/>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SampleComment">
        <xs:annotation>
          <xs:documentation>A general sample description</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType mixed="true">
          <xs:sequence>
            <xs:element ref="Validation" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="Value" type="xs:string"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="Validation">
        <xs:annotation>
          <xs:documentation xml:lang="en">

```

(continues on next page)

(continued from previous page)

A validation type which is an element/part of every other element in the schema. It is used to communicate validation issues as part of the output.

```

</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attribute name="IsValid" type="xs:boolean" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Indicates whether or not the element is valid. The assumption is that the
          Validation element is omitted unless the element is invalid, in which
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ID" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          An identifier which can be used by client applications to translate/map
          to a human decipherable message.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Source" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          This is the element which has experienced a validation issue.
        </xs:documentation>
      </xs:annotation>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="PlateId"/>
        <xs:enumeration value="PlateDefinition"
        </xs:enumeration>
        <xs:enumeration value="SchemaVersion"/
        </xs:enumeration>
        <xs:enumeration value="DefType"/>
        <xs:enumeration value="Owner"/>
        <xs:enumeration value="CreatedBy"/>
        <xs:enumeration value="Comments"/>
        <xs:enumeration value="OutputPath"/>
        <xs:enumeration value="Collections"/>
        <xs:enumeration value="Collection"/>
        <xs:enumeration value=
        </xs:enumeration>
        <xs:enumeration value="RunResources"/>
        <xs:enumeration value=
        </xs:enumeration>
        <xs:enumeration value="ChipLayout"/>
        <xs:enumeration value=
        </xs:enumeration>
        <xs:enumeration value="SequencingKit"/
        </xs:enumeration>
        <xs:enumeration value="RequiredTips"/>
        <xs:enumeration value=
        </xs:enumeration>
        <xs:enumeration value="EstimatedTotalRunTime"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:complexType>
</xs:documentation>

```

↪ schema. It is used to communicate validation issues as part of the output.

↪ case,

↪ "DNA Template Prep Kit Definition" />

↪ "Binding Kit Definition" />

↪ "Compatible Chip Layouts" />

↪ "Compatible Sequencing Kits" />

↪

↪ "EstimatedTotalRunTime" />

(continues on next page)

(continued from previous page)

```

    <xs:enumeration value=
    <xs:enumeration value=
    <xs:enumeration value="Basecaller"/>
    <xs:enumeration value=
    <xs:enumeration value="WellNo"/>
    <xs:enumeration value="SampleName"/>
    <xs:enumeration value="Barcode"/>
    <xs:enumeration value="AcquisitionTime
    <xs:enumeration value="InsertSize"/>
    <xs:enumeration value="ReuseComplex"/>
    <xs:enumeration value="StageHS"/>
    <xs:enumeration value=
    <xs:enumeration value="Confidence"/>
    <xs:enumeration value="SampleComment"/
    <xs:enumeration value="StrobeByTime"/>
  </xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="ElementPath" type="xs:string" use=
  <xs:annotation>
    <xs:documentation xml:lang="en">
      An optional string attribute which holds the path to the offending
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="SupplementalInfo" type="xs:string" use=
  <xs:annotation>
    <xs:documentation xml:lang="en">
      An optional string attribute which holds extraneous information.
    </xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:complexType name="AssayType">
  <xs:complexContent>
    <xs:extension base="pbbase:DataEntityType">
      <xs:sequence>
        <xs:element ref="SubreadSets"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ExperimentContainerType">
  <xs:annotation>
    <xs:documentation>A composite object type that can encompass
  </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element ref="SubreadSets"/>
  </xs:sequence>
</xs:complexType>

```

(continues on next page)

(continued from previous page)

One use case may be that a user may have a large genome they'd like to sequence, and it may take multiple runs on multiple instruments, to get enough data. Another use case may be that a user has multiple samples of the same phenotype which they would like to analyze in a similar fashion/automation, and as such these samples are run as part of one experiment.

The experiment object is intended to be packagable, such that the metadata of all acquisitions within is contained.

```

</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="pbbase:BaseEntityType">
      <xs:sequence>
        <xs:element name="InvestigatorName" type=
  "xs:string" minOccurs="0">
          <xs:annotation>
            <xs:documentation>An optional
  PI name</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="CreatedDate" type="xs:date">
          <xs:annotation>
            <xs:documentation>
  Automatically generated creation date</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Runs" minOccurs="0">
          <xs:annotation>
            <xs:documentation>Multiple
  acquisitions from different instrument runs</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Run
  " type="RunType" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="DataSets" minOccurs="0">
          <xs:annotation>
            <xs:documentation>Pointers to
  various data elements associated with the acquisitions</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element ref=
  "pbd:DataSet" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="RecordedEvents" minOccurs="0"
  ">
          <xs:annotation>
            <xs:documentation>Journal of
  metrics, system events, or alarms that were generated during this container's
  lifetime</xs:documentation>
          </xs:annotation>
        </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
          <xs:element name=
↳ "RecordedEvent" type="pbbase:RecordedEventType" minOccurs="0" maxOccurs="unbounded">
↳ <xs:annotation>
↳ <xs:documentation>Journal of metrics, system events, or alarms that were generated
↳ during this container's lifetime</xs:documentation>
          </
↳ <xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="BioSamples" minOccurs="0">
    <xs:complexType>
      <xs:sequence>
        <xs:element name=
↳ "BioSample" type="pbsample:BioSampleType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:attribute name="ExperimentId" type="xs:string"/>
<xs:attribute name="TimeStampedName" type="xs:string">
  <xs:annotation>
    <xs:documentation>This is NOT
↳ intended to be used as a unique field. For uniqueness, use UniqueId. In order to
↳ not utilize customer provided names, this attribute may be used as an alternative
↳ means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="RunType">
  <xs:annotation>
    <xs:documentation>A run is defined as a set of one or more
↳ data collections acquired in sequence on an instrument. A run specifies the wells
↳ and SMRT Cells to include in the sequencing run, along with the collection and
↳ analysis automation to use for the selected wells and cells.
  </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType">
      <xs:sequence>
        <xs:element name="Outputs" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref=
↳ "Output" minOccurs="0" maxOccurs="unbounded"/>
              <xs:element ref=
↳ "SubreadSets" minOccurs="0">
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
</xs:annotation>

```

```

↳ <xs:documentation>When a run definition is created, the placeholder for (continues on next page)
↳ is created as well. As part of that, the collection metadata will reside within
↳ this output placeholder.

```

(continued from previous page)

```

There will be one SubreadSet placeholder created per collection.</xs:documentation>
</
↳<xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Inputs" minOccurs="0">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Input
↳ " maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element ref="Assay" minOccurs="0">
  <xs:annotation>
    <xs:documentation>A
↳ predefined set of collection definitions for the purpose of conducting a known run
↳ type</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element ref="RunResources" minOccurs="0"/>
<xs:element name="RecordedEvents" minOccurs="0"
↳ ">
  <xs:annotation>
    <xs:documentation>Journal of
↳ metrics, system events, or alarms that were generated during this run's lifetime</
↳<xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=
↳ "RecordedEvent" type="pbbase:RecordedEventType" minOccurs="0" maxOccurs="unbounded">
↳<xs:annotation>
↳<xs:documentation>Journal of metrics, system events, or alarms that were generated
↳ during this run's lifetime.
In the case of Primary generating the DataSet containing the sts.xml, this
↳ RecordedEvent object should be a pointer to the DataSet object generated.</
↳<xs:documentation>
  </
↳<xs:annotation>
  </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Status" type=
↳ "pbbase:SupportedRunStates"/>
  <xs:attribute name="InstrumentId" type="xs:string">
    <xs:annotation>
      <xs:documentation>World unique id
↳ assigned by PacBio. </xs:documentation>
    </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        </xs:attribute>
        <xs:attribute name="InstrumentName" type="xs:string">
            <xs:annotation>
                <xs:documentation>Friendly name_
→ assigned by customer</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="CreatedBy" type="xs:string">
            <xs:annotation>
                <xs:documentation>Who created the run.
→ </xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="StartedBy" type="xs:string">
            <xs:annotation>
                <xs:documentation>Who started the run.
→ Could be different from who created it. </xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="WhenStarted">
            <xs:annotation>
                <xs:documentation>Date and time of_
→ when the overall run was started. </xs:documentation>
            </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:dateTime"/>
            </xs:simpleType>
        </xs:attribute>
    </xs:extension>
</xs:complexType>
</xs:complexType>
<xs:element name="CollectionReferences">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="CollectionMetadataRef" type=
→ "xs:IDREF" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SubreadSets">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="pbds:SubreadSet" maxOccurs="unbounded
→ "/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>

```

9.2 PacBioAutomationConstraints

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 4 sp1 (x64) (http://www.altova.com) by_
→ efayad@pacificbiosciences.com (Pacific Biosciences) -->

```

(continues on next page)

(continued from previous page)

```

<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioAutomationConstraints.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.org/2003/
  XInclude" xmlns:xm="http://www.w3.org/XML/1998/namespace" xmlns:pbbase="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" xmlns:pbrk="http://
  pacificbiosciences.com/PacBioReagentKit.xsd" xmlns:pbpn="http://pacificbiosciences.
  com/PacBioPartNumbers.xsd" targetNamespace="http://pacificbiosciences.com/
  PacBioAutomationConstraints.xsd" elementFormDefault="qualified" id=
  "PacBioAutomationConstraints">
  <xs:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation=
  "http://www.w3.org/2001/03/xml.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
  schemaLocation="PacBioReagentKit.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioPartNumbers.xsd"
  schemaLocation="PacBioPartNumbers.xsd"/>
  <xs:element name="PacBioAutomationConstraints" type=
  "PacBioAutomationConstraintsType">
    <xs:annotation>
      <xs:documentation>The root element of the Automation
    Constraints </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="AutomationConstraint" type="pbbase:AutomationConstraintType"
  />
  <xs:complexType name="PacBioAutomationConstraintsType">
    <xs:sequence>
      <xs:element name="AutomationConstraints">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="AutomationConstraint
  " maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element ref="pbpn:PacBioPartNumbers" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Version" type="xs:string"/>
    <xs:attribute name="Description" type="xs:string"/>
  </xs:complexType>
</xs:schema>

```

9.3 PacBioBaseDataModel

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
  efayad@pacificbiosciences.com (Pacific Biosciences) -->
<!-- W3C XML Schema generated by XMLSpy v2014 sp1 (x64) (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" targetNamespace="http://
  pacificbiosciences.com/PacBioBaseDataModel.xsd" elementFormDefault="qualified"
  attributeFormDefault="unqualified">

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="AnalogType">
  <xs:complexContent>
    <xs:extension base="BaseEntityType">
      <xs:sequence>
        <xs:element name="Spectrum">
          <xs:annotation>
            <xs:documentation>A vector of
↪probabilities, given in the order of increasing filter-bin wavelength, that light
↪emitted by the analog will fall in the corresponding filter bin of the instrument
↪detection system. By convention, the values are normalized to sum to 1.</
↪xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name=
↪"Values">
↪<xs:complexType>
↪<xs:sequence>
↪<xs:element name="Value" type="xs:float" maxOccurs="unbounded">
↪<xs:annotation>
↪<xs:documentation>There should be as many values as specified in the Number of
↪Filter Bins attribute.
Each value is a probability, in the range of [0, 1].</xs:documentation>
↪</xs:annotation>
↪</xs:element>
↪
↪</xs:sequence>
↪
↪</xs:complexType>
↪
↪</xs:element>
↪</xs:sequence>
↪<xs:attribute name=
↪"NumberFilterBins" type="xs:int" use="required">
↪
↪<xs:annotation>
↪<xs:documentation>number of bins describing the spectrum, green to red</
↪xs:documentation>
↪
↪</xs:annotation>
↪
↪</xs:attribute>
↪
↪</xs:complexType>
↪</xs:element>
↪<xs:element name="RelativeAmplitude" type=
↪"xs:float">
↪
↪<xs:annotation>
↪<xs:documentation>Relative
↪intensity of emission vs. a reference analog using standardized metrology - e.g.,
↪relative to the amplitude of the "542" analog as measured by the mean DWS pkMid on
↪the Astro instrument.</xs:documentation>
↪
↪</xs:annotation>
↪
↪</xs:element>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="IntraPulseXsnCV" type=
↪ "xs:float">
    <xs:annotation>
        <xs:documentation>The 1-sigma
↪ fractional variation of the intra-pulse signal, independent of any Shot noise
↪ associated with that signal</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="InterPulseXsnCV" type=
↪ "xs:float">
    <xs:annotation>
        <xs:documentation>The 1-sigma
↪ fractional variation, pulse-to-pulse, of the mean signal level (i.e., the pkMid).</
↪ xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="DiffusionXsnCV" type=
↪ "xs:float"/>
</xs:sequence>
<xs:attribute name="Base" type="SupportedNucleotides">
    <xs:annotation>
        <xs:documentation>The base label, A,
↪ C, T, or G</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="Nucleotide" type="xs:string">
    <xs:annotation>
        <xs:documentation>The type and number
↪ of nucleotides on a given analog. e.g. (dT6P)6</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="Wavelength" type="xs:float">
    <xs:annotation>
        <xs:documentation>The peak emission
↪ wavelength associated with the dye label in nm.</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="CompoundID" type="xs:string">
    <xs:annotation>
        <xs:documentation>Identification code
↪ of the final compound. The suffix 'N' should be used to distinguish these values
↪ from enzyme identifiers. e.g. 5031N</xs:documentation>
    </xs:annotation>
</xs:attribute>
<xs:attribute name="LotID" type="xs:string">
    <xs:annotation>
        <xs:documentation>Identification code
↪ for the build of the final compound, written as initials/date, where date is
↪ written as YYYY-MM-DD. e.g. js/2014-06-30</xs:documentation>
    </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="BaseEntityType">
    <xs:annotation>
        <xs:documentation>This is the base element type for all types
↪ in this data model</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

</xs:annotation>
<xs:sequence>
  <xs:element name="Extensions" minOccurs="0">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ExtensionElement"
minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:attribute name="Name" type="xs:string">
  <xs:annotation>
    <xs:documentation>A short text identifier; uniqueness
not necessary</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Description" type="xs:string">
  <xs:annotation>
    <xs:documentation>A long text description of the
object</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Tags" type="xs:string">
  <xs:annotation>
    <xs:documentation>A set of keywords assigned to the
object to help describe it and allow it to be found via search</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Format" type="xs:string">
  <xs:annotation>
    <xs:documentation>Optional, but recommended. The
MIME-Type of the referenced file. See http://www.iana.org/assignments/media-types/
media-types.xhtml for examples</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="ResourceId" type="xs:anyURI">
  <xs:annotation>
    <xs:documentation>A uniform resource identifier used
to identify a "web" resource. e.g. svc://run/acquisition/alignment/gridding</
xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Version" type="xs:string">
  <xs:annotation>
    <xs:documentation>An optional identifier denoting the
revision of this particular entity</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="CreatedAt">
  <xs:annotation>
    <xs:documentation>Timestamp designating the creation
of this object, relative to UTC; millisecond precision is expected.</
xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:simpleType>
  <xs:restriction base="xs:dateTime"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="ModifiedAt">
        <xs:annotation>
          <xs:documentation>Timestamp designating the
↪modification of this object, relative to UTC; millisecond precision is expected.</
↪xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  <xs:complexType name="StrictEntityType">
    <xs:annotation>
      <xs:documentation>This is the base element type for all types
↪in this data model</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="BaseEntityType">
        <xs:attribute name="UniqueId" use="required">
          <xs:annotation>
            <xs:documentation>A unique identifier,
↪such as a GUID - likely autogenerated</xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:ID">
              <xs:pattern value="[a-fA-F0-9]
↪{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="MetaType" type="xs:string" use=
↪"required">
          <xs:annotation>
            <xs:documentation>Controlled
↪Vocabulary, meant as a means to group similar entities; the type of the object, e.g.
↪Instrument Run, Secondary Run, Assay, Sample, Barcode, Alignment File, Alarm,
↪Exception, Metric, SystemEvent, etc.</xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="TimeStampedName" type="xs:string"
↪use="required">
          <xs:annotation>
            <xs:documentation>This is NOT
↪intended to be used as a unique field. For uniqueness, use UniqueId. In order to
↪not utilize customer provided names, this attribute may be used as an alternative
↪means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="ConfigSetAnalog">
    <xs:annotation>
      <xs:documentation>An unlimited number of analogs listed for
↪the purposes of hosting in a configuration file. e.g. a list of all possible
↪analogs on the system</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="AnalogType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="DataEntity" type="DataEntityType"/>
    <xs:complexType name="DataEntityType">
        <xs:annotation>
            <xs:documentation>Extends BaseEntityType and adds a value_
↪element. The intent is to have only one of the value elements exist at any point_
↪in time; however, this is not enforced.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="BaseEntityType">
                <xs:sequence>
                    <!--xs:element name="EncodedValue" type=
↪"xs:base64Binary" nillable="true" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>A complex_
↪data type element, such as an image, file, binary object, etc.</xs:documentation>
                        </xs:annotation>
                    </xs:element-->
                    <xs:element name="Checksum" type="xs:string"
↪minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>small-size_
↪datum of the attached value for the purpose of detecting errors or modification_
↪which may have been introduced during its transmission or storage</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="ValueDataType" type=
↪"SupportedDataTypes" default="Object">
                    <xs:annotation>
                        <xs:documentation>The datatype of the_
↪simple or encoded value. If not specified, a string is assumed.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="SimpleValue" type=
↪"xs:anySimpleType">
                    <xs:annotation>
                        <xs:documentation>A simple data type_
↪element, such as a string, int, float, etc.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MetaType" type="xs:string">
                    <xs:annotation>
                        <xs:documentation>Controlled_
↪Vocabulary, meant as a means to group similar entities; the type of the object, e.g.
↪Instrument Run, Secondary Run, Assay, Sample, Barcode, Alignment File, Alarm,_
↪Exception, Metric, SystemEvent, etc.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="TimeStampedName" type="xs:string">
                    <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>This is NOT
↳intended to be used as a unique field. For uniqueness, use UniqueId. In order to
↳not utilize customer provided names, this attribute may be used as an alternative
↳means of Human Readable ID, e.g. instrumentId-Run-150304_231155</xs:documentation>
                                </xs:annotation>
                                </xs:attribute>
                                </xs:extension>
                                </xs:complexContent>
</xs:complexType>
<xs:element name="DataPointers">
    <xs:annotation>
        <xs:documentation>Pointer list to UniqueIds in the system</
↳xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="DataPointer" type="xs:IDREF"
↳minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="DNABarcode">
        <xs:annotation>
            <xs:documentation>Composite of uuid, sequence, and name</
↳xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                <xs:extension base="BaseEntityType">
                    <xs:attribute name="DNASequence">
                        <xs:annotation>
                            <xs:documentation>This is the sample
↳'s DNA barcode</xs:documentation>
                        </xs:annotation>
                    </xs:attribute>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    <xs:element name="DyeSetAnalog">
        <xs:annotation>
            <xs:documentation>A set of four analogs, one for each of the
↳nucleotides, grouped together for the purposes of a single experiment.</
↳xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:complexContent>
                    <xs:extension base="AnalogType"/>
                </xs:complexContent>
            </xs:complexType>
        </xs:element>
    <xs:element name="ExternalResources">
        <xs:annotation>
            <xs:documentation>Pointers to data that do not reside inside
↳the parent structure</xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="ExternalResource" maxOccurs=
↳"unbounded"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:complexType name="InputOutputDataType">
      <xs:complexContent>
        <xs:extension base="StrictEntityType"/>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="IndexedDataType">
      <xs:complexContent>
        <xs:extension base="InputOutputDataType">
          <xs:sequence>
            <xs:element name="FileIndices" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name=
↪ "FileIndex" type="InputOutputDataType" maxOccurs="unbounded">

↪ <xs:annotation>

↪ <xs:documentation>e.g. index for output files, allowing one to find information in
↪ the output file</xs:documentation>

                  </
↪ <xs:annotation>

                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element ref="ExternalResources" minOccurs=
↪ "0"/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AutomationType">
      <xs:complexContent>
        <xs:extension base="BaseEntityType">
          <xs:sequence>
            <xs:element name="AutomationParameters"
↪ minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element ref=
↪ "AutomationParameter" maxOccurs="unbounded"/>

                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element ref="Defaults" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="PartNumber" type="xs:string" use=
↪ "optional">

            <xs:annotation>
              <xs:documentation>Defines a part
↪ number, mainly for use in defining incompatibility with other PB kit PNs, if
↪ necessary</xs:documentation>

            </xs:annotation>
          </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="IsRestricted" type="xs:boolean"
↪use="optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked for internal use or by admin users only</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="IsObsolete" type="xs:boolean" use=
↪"optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked as obsolete</xs:documentation>
            </xs:annotation>
        </xs:attribute>
        <xs:attribute name="IsDefault" type="xs:boolean" use=
↪"optional" default="false">
            <xs:annotation>
                <xs:documentation>Allows for an
↪automation to be marked as a default for a kit with which it's compatible</
↪xs:documentation>
            </xs:annotation>
        </xs:attribute>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:element name="AutomationParameter" type="DataEntityType">
    <xs:annotation>
        <xs:documentation>One or more collection parameters, such as
↪MovieLength, InsertSize, UseStageStart, IsControl, etc..</xs:documentation>
    </xs:annotation>
</xs:element>
<xs:complexType name="IncompatiblePairType">
    <xs:annotation>
        <xs:documentation>Describes a bidirectional incompatibility
↪between part numbers.

By default, any PN is compatible for use with other PNs in the system. In order to
↪exclude the usage of one or more PNs with this one, the pairwise incompatible PNs
↪are listed here.</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="BaseEntityType">
            <xs:attribute name="PartA" type="xs:string" use=
↪"required">
                <xs:annotation>
                    <xs:documentation>An automation or
↪kit Part Number that's incompatible with Part Number B</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="PartB" type="xs:string" use=
↪"required">
                <xs:annotation>
                    <xs:documentation>An automation or
↪kit Part Number that's incompatible with Part Number A</xs:documentation>
                </xs:annotation>
            </xs:attribute>
        </xs:extension>

```

(continues on next page)

(continued from previous page)

```

        </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="AutomationConstraintType">
        <xs:annotation>
          <xs:documentation>This data type defines constraints that an
↪automation has. The information here, along with the availability of an
↪exclusionary list of automations in the PartNumberType, allows for defining a
↪robust compatibility matrix.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="BaseEntityType">
            <xs:sequence>
              <xs:element name="Automations" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>Names of
↪automations that are all similarly constrained</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪"Automation" type="AutomationType" maxOccurs="unbounded"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute name="SupportsCellReuse" type=
↪"xs:boolean">
              <xs:annotation>
                <xs:documentation>Does this
↪automation support cell reuse?</xs:documentation>
              </xs:annotation>
            </xs:attribute>
            <xs:attribute name="SupportsStageStart" type=
↪"xs:boolean">
              <xs:annotation>
                <xs:documentation>Does this
↪automation support hot-start on the stage?</xs:documentation>
              </xs:annotation>
            </xs:attribute>
            <xs:attribute name="MaxCollectionsPerCell" type=
↪"xs:int">
              <xs:annotation>
                <xs:documentation>If cell reuse is
↪supported (i.e. above attribute is true) how many times can the cell be reused?</
↪xs:documentation>
              </xs:annotation>
            </xs:attribute>
            <xs:attribute name="MinMovieLength" type="xs:int">
              <xs:annotation>
                <xs:documentation>Minimum length of
↪movie acquisition</xs:documentation>
              </xs:annotation>
            </xs:attribute>
            <xs:attribute name="MaxMovieLength" type="xs:int">
              <xs:annotation>
                <xs:documentation>Maximum length of
↪movie acquisition</xs:documentation>
              </xs:annotation>
            </xs:attribute>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:schema>
  </pre>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="MinInsertSize" type="xs:int">
        <xs:annotation>
          <xs:documentation>Minimum recommended_
↪insert size</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="MaxInsertSize" type="xs:int">
        <xs:annotation>
          <xs:documentation>Maximum recommended_
↪insert size</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="PartNumberType">
  <xs:annotation>
    <xs:documentation>Generic representation of a supply kit.
    If the part number has an NFC associated with it, the contents of the NFC may be_
↪encoded here.</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataEntityType">
      <xs:sequence>
        <xs:element ref="Defaults" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="PartNumber" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit part number
↪</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="LotNumber" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit lot number</
↪xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="Barcode" type="xs:string">
        <xs:annotation>
          <xs:documentation>The kit barcode;_
↪used for tracking purposes.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="ExpirationDate" type="xs:date">
        <xs:annotation>
          <xs:documentation>The kit's shelf life
↪</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="IsObsolete" type="xs:boolean"_
↪default="false"/>
      <xs:attribute name="IsRestricted" type="xs:boolean"_
↪default="false"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="RecordedEventType">
      <xs:annotation>
        <xs:documentation>Metrics, system events, alarms, and logs_
↪may utilize this type</xs:documentation>
      </xs:annotation>
      <xs:complexContent>
        <xs:extension base="DataEntityType">
          <xs:attribute name="Context" type="xs:string">
            <xs:annotation>
              <xs:documentation>The part of the_
↪system in effect when the event was recorded</xs:documentation>
            </xs:annotation>
          </xs:attribute>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="SequencingChemistry">
      <xs:annotation>
        <xs:documentation>A container for a set of analogs</
↪xs:documentation>
      </xs:annotation>
      <xs:complexContent>
        <xs:extension base="DataEntityType">
          <xs:sequence>
            <xs:element name="DyeSet">
              <xs:complexType>
                <xs:complexContent>
                  <xs:extension base=
↪"BaseEntityType">
                    <xs:sequence>
                      <xs:element name="Analog">
                        <xs:complexType>
                          <xs:sequence>
                            <xs:element name="Analog" type="AnalogType" maxOccurs="4"/>
                          </xs:sequence>
                        </xs:complexType>
                      </xs:element>
                    </xs:sequence>
                  </xs:extension>
                </xs:complexContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:element name="PacBioSequencingChemistry">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>Root element for document containing the
↪ container of analog set, SequencingChemistryConfig</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="ChemistryConfig"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="SequencingChemistryConfig">
        <xs:annotation>
          <xs:documentation>A container for a set of analogs</
↪ xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="DataEntityType">
            <xs:sequence>
              <xs:element name="Analog">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪ "Analog" type="AnalogType" maxOccurs="4"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="DefaultLaserSetPoint" type=
↪ "xs:float">
                <xs:annotation>
                  <xs:documentation>The laser
↪ power at the input couple, needed to achieve predefined performance requirements
↪ based on a median 'golden' SMRT Cell.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="SNRCut" type="xs:float">
                <xs:annotation>
                  <xs:documentation>The SNRCut
↪ is applied in PPA (baz2bam) as a read-quality filter.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="TargetSNR">
                <xs:complexType>
                  <xs:attribute name="SNR_A"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_C"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_G"
↪ type="xs:float" use="required"/>
                  <xs:attribute name="SNR_T"
↪ type="xs:float" use="required"/>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:complexType name="StatsContinuousDistType">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>Continuous distribution class</
↪xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="BaseEntityType">
            <xs:sequence>
              <xs:element name="SampleSize" type="xs:int"/>
              <xs:element name="SampleMean" type="xs:float"/
↪>
              <xs:element name="SampleMed" type="xs:float"/>
              <xs:element name="SampleStd" type="xs:float"/>
              <xs:element name="Sample95thPct" type=
↪"xs:float"/>
              <xs:element name="NumBins" type="xs:int"/>
              <xs:element name="BinCounts">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪"BinCount" type="xs:int" maxOccurs="unbounded"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="BinWidth" type="xs:float"/>
              <xs:element name="MinOutlierValue" type=
↪"xs:float" minOccurs="0"/>
              <xs:element name="MinBinValue" type="xs:float"
↪" minOccurs="0"/>
              <xs:element name="MaxBinValue" type="xs:float"
↪" minOccurs="0"/>
              <xs:element name="MaxOutlierValue" type=
↪"xs:float" minOccurs="0"/>
              <xs:element name="MetricDescription" type=
↪"xs:string"/>
            </xs:sequence>
            <xs:attribute name="Channel" type="xs:string"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="StatsDiscreteDistType">
        <xs:annotation>
          <xs:documentation>Discrete distribution class</
↪xs:documentation>
        </xs:annotation>
        <xs:complexContent>
          <xs:extension base="BaseEntityType">
            <xs:sequence>
              <xs:element name="NumBins" type="xs:int"/>
              <xs:element name="BinCounts">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name=
↪"BinCount" type="xs:int" maxOccurs="unbounded"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="MetricDescription" type=
↪"xs:string"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:element name="BinLabels">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name=
↪ "BinLabel" type="xs:string" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="StatsTimeSeriesType">
    <xs:annotation>
        <xs:documentation>Time series (for time-dependent metrics)</
↪ xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="BaseEntityType">
            <xs:sequence>
                <xs:element name="TimeUnits" type="xs:string"/
↪ >
                <xs:element name="ValueUnits" type="xs:string
↪ "/>
                <xs:element name="StartTime" type="xs:float"/>
                <xs:element name="MeasInterval" type="xs:float
↪ "/>
                <xs:element name="Values" minOccurs="0">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Val
↪ " type="xs:float" minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="SupplyKitBinding">
    <xs:annotation>
        <xs:documentation>A more specific binding kit representation
↪ (includes SupplyKit fields). </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
        <xs:extension base="PartNumberType">
            <xs:sequence>
                <xs:element name="Control" type=
↪ "SupplyKitControl" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>Defines the
↪ binding kit internal control name. Present when used, otherwise not used if not
↪ defined. </xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="IsControlUsed" type=
↪ "xs:boolean" minOccurs="0">

```

(continues on next page)

(continued from previous page)

```

                                <xs:annotation>
                                    <xs:documentation>True if the
↪control was used during run, otherwise false. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitCellPack">
                <xs:annotation>
                    <xs:documentation>Represents the package of cells. </
↪xs:documentation>
                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="PartNumberType">
                        <xs:sequence>
                            <xs:element name="ChipLayout" type="xs:string
↪" minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>Defines the
↪internal chip layout name, if any. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="SupportsCellReuse">
                            <xs:annotation>
                                <xs:documentation>If
↪SupportsCellReuse is true, it can be used for regular sequencing as well as in a
↪reuse scenario.</xs:documentation>
                            </xs:annotation>
                        </xs:attribute>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitControl">
                <xs:annotation>
                    <xs:documentation>Represents the DNA control complex. </
↪xs:documentation>
                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="PartNumberType">
                        <xs:sequence>
                            <xs:element name="InternalControlName" type=
↪"xs:string" minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>Defines the
↪internal control name, if any. </xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="CustomSequence" type=
↪"xs:string" nillable="true" minOccurs="0"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SupplyKitTemplate">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
            <xs:documentation>A more specific template kit representation
↳ (includes SupplyKit fields). </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="PartNumberType">
                <xs:sequence>
                    <xs:element name="LeftAdaptorSequence" type=
↳ "xs:string" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>Left
↳ adapter DNA sequence.</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LeftPrimerSequence" type=
↳ "xs:string" minOccurs="0"/>
                    <xs:element name="RightAdaptorSequence" type=
↳ "xs:string" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>Right
↳ adapter DNA sequence. If not specified, a symmetric adapter model is inferred,
↳ where the left adapter sequence is used wherever needed.</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="RightPrimerSequence" type=
↳ "xs:string" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>Right
↳ primer sequence. If not specified, a symmetric model is inferred, where the left
↳ primer sequence is used wherever needed.</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="MinInsertSize" type="xs:int">
                    <xs:annotation>
                        <xs:documentation>Minimum recommended
↳ insert size</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute name="MaxInsertSize" type="xs:int">
                    <xs:annotation>
                        <xs:documentation>Maximum recommended
↳ insert size</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:simpleType name="SupportedAcquisitionStates">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Ready"/>
            <xs:enumeration value="Initializing"/>
            <xs:enumeration value="SocketDiagnostics"/>
            <xs:enumeration value="Acquiring"/>
            <xs:enumeration value="Aligning"/>
            <xs:enumeration value="Aligned"/>
            <xs:enumeration value="Aborting"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:enumeration value="Aborted"/>
        <xs:enumeration value="Failed"/>
        <xs:enumeration value="Completing"/>
        <xs:enumeration value="Complete"/>
        <xs:enumeration value="Calibrating"/>
        <xs:enumeration value="Unknown"/>
        <xs:enumeration value="Pending"/>
        <xs:enumeration value="ReadyToCalibrate"/>
        <xs:enumeration value="CalibrationComplete"/>
        <xs:enumeration value="ReadyToAcquire"/>
        <xs:enumeration value="FinishingAnalysis"/>
        <xs:enumeration value="PostPrimaryPending"/>
        <xs:enumeration value="PostPrimaryAnalysis"/>
        <xs:enumeration value="TransferPending"/>
        <xs:enumeration value="TransferringResults"/>
        <xs:enumeration value="Error"/>
        <xs:enumeration value="Stopped"/>
        <xs:enumeration value="TransferFailed"/>
        <xs:enumeration value="InPrep"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedDataTypes">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Int16"/>
        <xs:enumeration value="Int32"/>
        <xs:enumeration value="Int64"/>
        <xs:enumeration value="UInt16"/>
        <xs:enumeration value="UInt32"/>
        <xs:enumeration value="UInt64"/>
        <xs:enumeration value="Boolean"/>
        <xs:enumeration value="Single"/>
        <xs:enumeration value="Double"/>
        <xs:enumeration value="String"/>
        <xs:enumeration value="DateTime"/>
        <xs:enumeration value="Int16_1D"/>
        <xs:enumeration value="Int32_1D"/>
        <xs:enumeration value="Int64_1D"/>
        <xs:enumeration value="UInt16_1D"/>
        <xs:enumeration value="UInt32_1D"/>
        <xs:enumeration value="UInt64_1D"/>
        <xs:enumeration value="Boolean_1D"/>
        <xs:enumeration value="Single_1D"/>
        <xs:enumeration value="Double_1D"/>
        <xs:enumeration value="String_1D"/>
        <xs:enumeration value="DateTime_1D"/>
        <xs:enumeration value="Int16_2D"/>
        <xs:enumeration value="Int32_2D"/>
        <xs:enumeration value="Int64_2D"/>
        <xs:enumeration value="UInt16_2D"/>
        <xs:enumeration value="UInt32_2D"/>
        <xs:enumeration value="UInt64_2D"/>
        <xs:enumeration value="Boolean_2D"/>
        <xs:enumeration value="Single_2D"/>
        <xs:enumeration value="Double_2D"/>
        <xs:enumeration value="String_2D"/>
        <xs:enumeration value="DateTime_2D"/>
        <xs:enumeration value="XML"/>
    </xs:restriction>
</xs:simpleType>

```

(continues on next page)

(continued from previous page)

```

        <xs:enumeration value="JSON"/>
        <xs:enumeration value="Object"/>
        <xs:enumeration value="Other"/>
        <xs:enumeration value="Unknown"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedNucleotides">
    <xs:restriction base="xs:string">
        <xs:enumeration value="A"/>
        <xs:enumeration value="C"/>
        <xs:enumeration value="T"/>
        <xs:enumeration value="G"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="SupportedRunStates">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Ready"/>
        <xs:enumeration value="Idle"/>
        <xs:enumeration value="System Test"/>
        <xs:enumeration value="Starting"/>
        <xs:enumeration value="Running"/>
        <xs:enumeration value="Aborting"/>
        <xs:enumeration value="Aborted"/>
        <xs:enumeration value="Terminated"/>
        <xs:enumeration value="Completing"/>
        <xs:enumeration value="Complete"/>
        <xs:enumeration value="Paused"/>
        <xs:enumeration value="Unknown"/>
    </xs:restriction>
</xs:simpleType>
<xs:complexType name="UserDefinedFieldsType">
    <xs:sequence>
        <xs:element name="DataEntities" type="DataEntityType"
↳maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="ValueDataType" type="SupportedDataTypes"/>
<!-- The base supply kit definition -->
<xs:element name="ExtensionElement" type="xs:anyType">
    <xs:annotation>
        <xs:documentation>A generic element whose contents are
↳undefined at the schema level. This is used to extend the data model.</
↳xs:documentation>
    </xs:annotation>
</xs:element>
<xs:complexType name="FilterType">
    <xs:sequence>
        <xs:element name="Properties">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="Property" maxOccurs=
↳"unbounded">
                        <xs:complexType>
                            <xs:attribute name=
↳"Name" type="xs:string" use="required"/>
                            <xs:attribute name=
↳"Value" type="xs:string" use="required"/>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name=
↪ "Operator" type="xs:string" use="required"/>
<xs:attribute name=
↪ "Feature" type="xs:string" use="optional"/>
<xs:attribute name=
↪ "Assignment" type="xs:string" use="optional"/>
        </xs:complexType>
    </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:element name="ExternalResource">
    <xs:annotation>
        <xs:documentation>for example, an output file could be the_
↪ BAM file, which could be associated with multiple indices into it.</
↪ xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:complexContent>
            <xs:extension base="IndexedDataType"/>
        </xs:complexContent>
    </xs:complexType>
</xs:element>
<xs:element name="KeyValueMap">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Items">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="Item" type=
↪ "MapItemType" minOccurs="0" maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:complexType>
                <xs:unique name="Item">
                    <xs:selector xpath="Item"/>
                    <xs:field xpath="Key"/>
                </xs:unique>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:complexType name="MapType">
    <xs:sequence>
        <xs:element ref="KeyValueMap"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="MapItemType">
    <xs:sequence>
        <xs:element name="Key" type="xs:ID"/>
        <xs:element name="Value" type="xs:anyType"/>
        <xs:element name="Description" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="ChemistryConfig" type="SequencingChemistryConfig"/>
<xs:element name="Defaults" type="DefaultsType" nillable="true">

```

(continues on next page)

(continued from previous page)

```

<xs:annotation>
  <xs:documentation>Default paramaters and filters which may be applied to
  ↳PartNumber types in order to constrain them via parameterization</xs:documentation>
</xs:annotation>
</xs:element>
<xs:complexType name="DefaultsType">
  <xs:annotation>
    <xs:documentation>A data type that allows the definition of default
    ↳paramaters and filters. This structure may be applied to PartNumber types in order
    ↳to constrain them via parameterization</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="AutomationParameters" nillable="true" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="AutomationParameter" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Filters" nillable="true" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Filter" type="FilterType" maxOccurs=
          ↳"unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:schema>

```

9.4 PacBioCollectionMetadata

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2016 (x64) (http://www.altova.com) by
  ↳efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="application/xml" href="metadata2html.xslt"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  ↳xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.
  ↳com/PacBioBaseDataModel.xsd" xmlns:pbsample="http://pacificbiosciences.com/
  ↳PacBioSampleInfo.xsd" xmlns:pbrk="http://pacificbiosciences.com/PacBioReagentKit.xsd
  ↳" targetNamespace="http://pacificbiosciences.com/PacBioCollectionMetadata.xsd"
  ↳elementFormDefault="qualified" id="PacBioCollectionMetadata">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  ↳schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
  ↳schemaLocation="PacBioSampleInfo.xsd"/>
  <xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
  ↳schemaLocation="PacBioReagentKit.xsd"/>
  <!-- The root element for the metadata structure -->
  <xs:element name="PacBioCollectionMetadata">
    <xs:annotation>
      <xs:documentation>Root element of a standalone
      ↳CollectionMetadata file.</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="CollectionMetadata"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Collections">
        <xs:annotation>
            <xs:documentation>A set of acquisition definitions</
->xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="CollectionMetadata" maxOccurs=
->"unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="CollectionMetadata">
        <xs:annotation>
            <xs:documentation>Root-level element for the metadata. The
->purpose of which is to contain pertinent instrument information related to the
->conditions present during a movie acquisition. It also serves to provide key
->pieces of information for integration with primary and secondary analysis. This
->file is associated with 1 movie. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="pbbase:StrictEntityType">
                    <xs:sequence>
                        <xs:element name="InstCtrlVer" type=
->"xs:string" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
->Instrument control software version. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element name="SigProcVer" type=
->"xs:string" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
->Signal processing software version. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element ref="RunDetails"
->minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
->Container for instrument run-related information. </xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element ref="Movie" minOccurs="0">
                            <xs:annotation>
                                <xs:documentation>
->Container for movie related information. This is populated by primary analysis.</
->xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element ref="WellSample">
        <xs:annotation>
          <xs:documentation>
↪ Container for sample related information, specific to the plate well. This could
↪ be a pooled sample with many barcodes, associated with many biological samples.

If multiple collections are utilizing the same physical plate well, all these
↪ collections should share the same WellSample UniqueId.</xs:documentation>
          </xs:annotation>
        </xs:element>
      <xs:element name="Automation" type=
↪ "pbbase:AutomationType">

        <xs:annotation>
          <xs:documentation>
↪ Defines the collection workflow (e.g., robotic movement, movie acquisition) for a
↪ particular cell. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CollectionNumber"
↪ type="xs:int" minOccurs="0">

        <xs:annotation>
          <xs:documentation>
↪ Collection number for this plate well. Sample from one plate well or tube can be
↪ distributed to more than one cell. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellIndex" type=
↪ "xs:int" minOccurs="0">

        <xs:annotation>
          <xs:documentation>The
↪ zero-based index of this particular cell within the cell tray. Likely to be in the
↪ range of [0-3]</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SetNumber" type=
↪ "xs:unsignedShort" minOccurs="0">

        <xs:annotation>
          <xs:documentation>
↪ Formerly known as the look number. 1 - N. Defaults to 1. 0 if the look is unknown.
↪ </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellPac" type=
↪ "pbbase:SupplyKitCellPack" minOccurs="0">

        <xs:annotation>
          <xs:documentation>The
↪ SMRT cell packaging supply information. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="ControlKit" type=
↪ "pbbase:SupplyKitControl" minOccurs="0">

        <xs:annotation>
          <xs:documentation>
↪ Defines the DNA control used for this experiment. </xs:documentation>
        </xs:annotation>
      </xs:element>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<xs:element name="TemplatePrepKit"
↳type="pbbase:SupplyKitTemplate" minOccurs="0">
    <xs:annotation>
        <xs:documentation>
↳Defines the template (sample) prep kit used for this experiment. Can be used to get
↳back to the primary and adapter used. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="BindingKit" type=
↳"pbbase:SupplyKitBinding" minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳binding kit supply information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="SequencingKitPlate"
↳type="pbrk:SupplyKitSequencing" minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳sequencing kit supply information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="Primary" minOccurs="0
↳">
    <xs:annotation>
        <xs:documentation>The
↳primary analysis-specific information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="Secondary" minOccurs=
↳"0">
    <xs:annotation>
        <xs:documentation>The
↳secondary analysis-specific information. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="UserDefinedFields"
↳minOccurs="0" maxOccurs="1">
    <xs:annotation>
        <xs:documentation>A
↳set of custom key-value pairs, set by a user when building a run definition. </
↳xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element ref="ExpirationData"
↳minOccurs="0">
    <xs:annotation>
        <xs:documentation>The
↳information on whether consumables were expired. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="ComponentVersions"
↳nillable="true" minOccurs="0">
    <xs:annotation>
        <xs:documentation>
↳Subcomponents involved in the generation of the data</xs:documentation>

```

(continues on next page)

(continued from previous page)

```

</xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element
↳name="VersionInfo" type="pbbase:BaseEntityType" maxOccurs="unbounded">

↳<xs:annotation>

↳<xs:documentation>Each component should list its name and version attribute</
↳xs:documentation>

</
↳xs:annotation>

</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Context" type="xs:string">
  <xs:annotation>
    <xs:documentation>Replace
↳with TimeStampedName</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="Status" type=
↳"pbbase:SupportedAcquisitionStates"/>
  <xs:attribute name="InstrumentId" type=
↳"xs:string">
    <xs:annotation>
      <xs:documentation>World
↳unique id assigned by PacBio. </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="InstrumentName" type=
↳"xs:string">
    <xs:annotation>
      <xs:documentation>Friendly
↳name assigned by customer</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<xs:element name="RunDetails">
  <xs:annotation>
    <xs:documentation>Information related to an instrument run.
↳A run can contain multiple chips, wells, and movies. </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TimeStampedName" type="xs:string">
        <xs:annotation>
          <xs:documentation>A unique identifier
↳for this run. Format is r[sid]_[iname]_[ts]. Where [id] is a system generated id
↳and [iname] is the instrument name and [ts] is a timestamp YYMMDD Example: r000123_
↳00117_100713 </xs:documentation>
        </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<xs:element name="Name" type="xs:string" minOccurs="0"
↳ ">
    <xs:annotation>
        <xs:documentation>Assigned name for a
↳ run, which consists of multiple wells. There is no constraint on the uniqueness of
↳ this data. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="CreatedBy" type="xs:string"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Who created the run.
↳ </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="WhenCreated" type="xs:dateTime"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Date and time of
↳ when the overall run was created in the system. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="StartedBy" type="xs:string"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Who started the run.
↳ Could be different from who created it. </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="WhenStarted" type="xs:dateTime"
↳ minOccurs="0">
    <xs:annotation>
        <xs:documentation>Date and time of
↳ when the overall run was started. </xs:documentation>
    </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Movie">
    <xs:annotation>
        <xs:documentation>A movie corresponds to one acquisition for
↳ a chip, one set (look) and one strobe. </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="WhenStarted" type="xs:dateTime">
                <xs:annotation>
                    <xs:documentation>Date and time of
↳ when this movie acquisition started. </xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="DurationInSec" type="xs:int"
↳ default="0">
                <xs:annotation>
                    <xs:documentation>The actual length
↳ of the movie acquisition (in seconds), irrespective of the movie duration specified
↳ by an automation parameter. </xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element name="Number" type="xs:int" default="0">
        <xs:annotation>
          <xs:documentation>The number of this_
↪movie within the set (i.e., look). This is unique when combined with the 'SetNumber
↪'. </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="ExpirationData">
  <xs:annotation>
    <xs:documentation>Container for the expired consumable data.
↪</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TemplatePrepKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the template prep kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="BindingKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the binding kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellPacPastExpiration" type="xs:int
↪">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the cell pac was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CellPacPastExpiration" type="xs:int
↪">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the eight pac was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SequencingKitPastExpiration" type=
↪"xs:int">
        <xs:annotation>
          <xs:documentation>Number of days past_
↪expiration the reagent kit was (if at all). </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SequencingTube0PastExpiration" type=
↪"xs:int">
        <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>Number of days past
↪expiration the reagent tube 0 was (if at all). </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="SequencingTubelPastExpiration" type=
↪"xs:int">
                                <xs:annotation>
                                <xs:documentation>Number of days past
↪expiration the reagent tube 1 was (if at all). </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                </xs:sequence>
                                </xs:complexType>
                                </xs:element>
                                <xs:element name="WellSample">
                                <xs:annotation>
                                <xs:documentation>Container for the sample related data. </
↪xs:documentation>
                                </xs:annotation>
                                <xs:complexType>
                                <xs:complexContent>
                                <xs:extension base="pbbase:BaseEntityType">
                                <xs:sequence>
                                <xs:element name="WellName" type=
↪"xs:string">
                                <xs:annotation>
                                <xs:documentation>
↪Identifies which well this sample came from (e.g., coordinate on a plate). </
↪xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="Concentration" type=
↪"xs:double">
                                <xs:annotation>
                                <xs:documentation>
↪Sample input concentration. </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="InsertSize" type=
↪"xs:int">
                                <xs:annotation>
                                <xs:documentation>
↪Length of the sheared template, e.g. 500, 2000, 30000</xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="SampleReuseEnabled"
↪type="xs:boolean">
                                <xs:annotation>
                                <xs:documentation>
↪Whether or not complex reuse is enabled for this well. </xs:documentation>
                                </xs:annotation>
                                </xs:element>
                                <xs:element name="StageHotstartEnabled
↪" type="xs:boolean">
                                <xs:annotation>
                                <xs:documentation>
↪Whether or not hotstart at the stage is enabled for this well. </xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element name="SizeSelectionEnabled"
↪ " type="xs:boolean">
        <xs:annotation>
          <xs:documentation>
↪ Whether or not size selection is enabled for this well. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="UseCount" type="
↪ "xs:int">
        <xs:annotation>
          <xs:documentation>
↪ Count of usages for this batch of complex. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="DNAControlComplex"
↪ type="xs:string" minOccurs="0">
        <xs:annotation>
          <xs:documentation>
↪ Indicating what kind (if any) control was used. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="SampleBarcodeInfo"
↪ type="pbbase:DataEntityType" minOccurs="0">
        <xs:annotation>
          <xs:documentation>
↪ When utilizing DNA barcoding, store the list of sample barcodes in this element.</
↪ xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element ref="
↪ "pbsample:BioSamplePointers" minOccurs="0">
        <xs:annotation>
          <xs:documentation>
↪ Back references to other sample object UniqueIds
This is optional since we will likely not have a link back to samples created in the
↪ calculator.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>
</xs:element>
<!-- Primary analysis-related fields.-->
<xs:element name="Primary">
  <xs:annotation>
    <xs:documentation>Container for the primary analysis related
↪ data. </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SampleTrace" minOccurs="0"
↪ maxOccurs="1">
        <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

                                <xs:documentation>Tag to indicate_
↪that the trace file will be sampled. </xs:documentation>
                                </xs:annotation>
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name=
↪"TraceSamplingFactor" type="xs:float">
                                            <xs:annotation>
↪<xs:documentation>Percentage of traces to sample. </xs:documentation>
                                                </xs:annotation>
                                            </xs:element>
                                        <xs:element name=
↪"FullPulseFile" type="xs:boolean">
                                            <xs:annotation>
↪<xs:documentation>Whether full or sampled pulse file is transferred if requested. </
↪xs:documentation>
                                                </xs:annotation>
                                            </xs:element>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                                <xs:element name="AutomationName" type="xs:string">
                                    <xs:annotation>
                                        <xs:documentation>Name of primary_
↪analysis protocol. </xs:documentation>
                                    </xs:annotation>
                                </xs:element>
                                <xs:element name="ConfigFileName" type="xs:string">
                                    <xs:annotation>
                                        <xs:documentation>Name of primary_
↪analysis config file. </xs:documentation>
                                    </xs:annotation>
                                </xs:element>
                                <xs:element name="SequencingCondition" type="xs:string
↪">
                                    <xs:annotation>
                                        <xs:documentation>A sequencing_
↪condition tag to be used by primary analysis, e.g., to select basecaller_
↪calibration or training parameters. </xs:documentation>
                                    </xs:annotation>
                                </xs:element>
                                <xs:element name="OutputOptions">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name=
↪"ResultsFolder" type="xs:string">
                                                <xs:annotation>
↪<xs:documentation>NOTE: not for customers. A sub-folder under the CollectionPath_
↪created by Primary Analysis. This is a field that will be updated by the primary_
↪analysis pipeline. The default (as created by homer) should be set to 'Reports_Sms
↪' for now. Consumers of the data should be aware that they will find collection_
↪metadata (and trace files if acquisition is so-configured) at the CollectionPathUri,
↪and all primary analysis results in the sub-folder PrimaryResultsFolder. </
↪xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element name=
↪ "CollectionPathUri" type="xs:anyURI">
        <xs:annotation>
↪ <xs:documentation>User-specified location of where the results should be copied,
↪ after an analysis has been completed. </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="CopyFiles">
        <xs:complexType>
          <xs:sequence>
↪ <xs:element name="CollectionFileCopy" type="PapOutputFile" maxOccurs="unbounded">
↪ <xs:annotation>
↪ <xs:documentation>Defines the set of files to be copied to the CollectionPathUri. 1
↪ or more. </xs:documentation>
↪ </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Readout">
    <xs:annotation>
↪ <xs:documentation>BazIO Readout option; valid values are Bases (default) and Pulses
↪ </xs:documentation>
    </xs:annotation>
    <xs:simpleType>
↪ <xs:restriction base="xs:string">
↪ <xs:enumeration value="Pulses"/>
↪ <xs:enumeration value="Bases"/>
↪ <xs:enumeration value="Bases_Without_QVs"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
<xs:element name=
↪ "MetricsVerbosity" default="Minimal">
    <xs:annotation>
↪ <xs:documentation>BazIO MetricsVerbosity option; valid values are Minimal (default),
↪ High, and None</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
↪ <xs:restriction base="xs:string">

```

(continues on next page)

(continued from previous page)

```

↪<xs:enumeration value="Minimal"/>
↪<xs:enumeration value="High"/>
↪<xs:enumeration value="None"/>
↪</xs:restriction>
↪</xs:simpleType>
↪</xs:element>
↪<xs:element name="
↪"TransferResource" minOccurs="0">
↪<xs:annotation>
↪<xs:documentation>Transfer Resource (optional for now, but will be made required_
↪when ICS implements this)</xs:documentation>
↪</xs:annotation>
↪<xs:complexType>
↪<xs:sequence>
↪<xs:element name="Id">
↪<xs:annotation>
↪<xs:documentation>Id of the Transfer Resource that is unique to the Scheme Type. A_
↪tuple of (TransferScheme, Id) will be globally unique</xs:documentation>
↪</xs:annotation>
↪<xs:simpleType>
↪<xs:restriction base="xs:string">
↪<xs:pattern value="([a-zA-Z0-9_\-]) *"/>
↪</xs:restriction>
↪</xs:simpleType>
↪</xs:element>
↪<xs:element name="TransferScheme">
↪<xs:annotation>
↪<xs:documentation>Transfer Scheme type (this should be an enum Scheme of rsync, srs_
↪or nfs)</xs:documentation>
↪</xs:annotation>
↪<xs:simpleType>
↪<xs:restriction base="xs:string">
↪<xs:enumeration value="RSYNC"/>
↪<xs:enumeration value="SRS"/>

```

(continues on next page)

(continued from previous page)

```

↪<xs:enumeration value="NFS"/>
↪</xs:restriction>
↪</xs:simpleType>
↪xs:element>
↪<xs:element name="Name" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Display Name of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="Description" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Description of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="DestPath" type="xs:string">
↪<xs:annotation>
↪<xs:documentation>Remote Root Destination Path of the Transfer Resource</
↪xs:documentation>
↪</xs:annotation>
↪xs:element>
↪<xs:element name="RelativePath" type="xs:string" minOccurs="0">
↪<xs:annotation>
↪<xs:documentation>Remote Relative Path of the Transfer Resource</xs:documentation>
↪</xs:annotation>
↪xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>

```

(continues on next page)

(continued from previous page)

```

</xs:element>
<!-- Secondary Analysis-related fields.-->
<xs:element name="Secondary">
  <xs:annotation>
    <xs:documentation>Container for the primary analysis related
↳data. </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AutomationName" type="xs:string">
        <xs:annotation>
          <xs:documentation>The secondary
↳analysis protocol name specified in the sample sheet. Ignored by secondary. </
↳xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="AutomationParameters" minOccurs="0">
        <xs:annotation>
          <xs:documentation>The parameters for
↳secondary analysis specified in the sample sheet. Ignored by secondary. </
↳xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name=
↳"AutomationParameter" type="pbbase:DataEntityType" minOccurs="0" maxOccurs=
↳"unbounded">
              <xs:annotation>
                <xs:documentation>One or more secondary analysis parameters, such as JobName,
↳Workflow, etc..</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="CellCountInJob" type="xs:int">
        <xs:annotation>
          <xs:documentation>The number of cells
↳in this secondary analysis job, identified by the secondary analysis parameter
↳'JobName'. Supports automated secondary analysis. </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Contains custom key/value pairs defined by the customer -->
<xs:element name="UserDefinedFields" type="pbbase:UserDefinedFieldsType">
  <xs:annotation>
    <xs:documentation>A set of key-value pairs specified by a
↳user via the run input mechanism. Note that uniqueness of keys is not enforced here,
↳and so may contain duplicate keys. </xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="KeyValue">
  <xs:annotation>
    <xs:documentation>One custom, possibly non-unique, key-value
↳pair. </xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="key" type="xs:string" use=
↪ "required">
                        <xs:annotation>
                            <xs:documentation>Key
↪ (attribute) and Value (element content). </xs:documentation>
                        </xs:annotation>
                    </xs:attribute>
                    <xs:attribute name="label" type="xs:string"
↪ use="optional"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:simpleType name="PapOutputFile">
        <xs:annotation>
            <xs:documentation>Defines a list of available file output
↪ types from primary output that can be copied out to the CollectionPathUri. The
↪ types Pulse, Base, Fasta, and Fastq are for legacy use only.</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="None"/>
            <xs:enumeration value="Trace"/>
            <xs:enumeration value="Fasta"/>
            <xs:enumeration value="Baz"/>
            <xs:enumeration value="Bam"/>
            <xs:enumeration value="DarkFrame"/>
            <xs:enumeration value="StatsH5"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>

```

9.5 PacBioDatasets

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2015 rel. 3 sp1 (x64) (http://www.altova.com) by Lakhvir Rai
↪ (Pacific Biosciences) -->
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioDatasets.xsd" xmlns:xs="http://
↪ www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↪ PacBioBaseDataModel.xsd" xmlns:pbmeta="http://pacificbiosciences.com/
↪ PacBioCollectionMetadata.xsd" xmlns:pbsample="http://pacificbiosciences.com/
↪ PacBioSampleInfo.xsd" targetNamespace="http://pacificbiosciences.com/PacBioDatasets.
↪ xsd" elementFormDefault="qualified" id="PacBioDatasets">
    <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↪ schemaLocation="PacBioBaseDataModel.xsd"/>
    <xs:import namespace="http://pacificbiosciences.com/PacBioCollectionMetadata.
↪ xsd" schemaLocation="PacBioCollectionMetadata.xsd"/>
    <xs:import namespace="http://pacificbiosciences.com/PacBioSampleInfo.xsd"
↪ schemaLocation="PacBioSampleInfo.xsd"/>
    <xs:element name="AlignmentSet">

```

(continues on next page)

(continued from previous page)

```

        <xs:annotation>
          <xs:documentation>DataSets for aligned subreads and CCS reads.
↪</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AlignmentSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="BarcodeSet">
        <xs:annotation>
          <xs:documentation>DataSets of Barcodes. Basically a thin_
↪metadata layer on top of the barcode FASTA.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="BarcodeSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="ConsensusAlignmentSet">
        <xs:annotation>
          <xs:documentation>DataSets of aligned CCS reads.</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AlignmentSetType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="ConsensusReadSet">
        <xs:annotation>
          <xs:documentation>DataSets of CCS reads (typically in_
↪unaligned BAM format).</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="DataSetType">
              <xs:sequence>
                <xs:element name="DataSetMetadata" _
↪type="DataSetMetadataType"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="Contigs">
        <xs:annotation>
          <xs:documentation>List of contigs in a ContigSet</
↪xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="pbbase:BaseEntityType">
              <xs:sequence>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Contig" minOccurs="0"
  " maxOccurs="unbounded">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension
          base="pbbase:BaseEntityType">
            <xs:attribute name="Length" use="required"/>
            <xs:attribute name="Digest" use="required"/>
          </
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:extension>
</xs:complexType>
</xs:element>
<xs:element name="ContigSet">
  <xs:annotation>
    <xs:documentation>DataSets of contigs sequences. Basically a
    thin metadata layer on top of a contigs FASTA (e.g. from HGAP).</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ContigSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="DataSet" type="DataSetType"/>
<xs:element name="DataSetRoot" type="DataSetRootType"/>
<xs:element name="HdfSubreadSet">
  <xs:annotation>
    <xs:documentation>DataSets of subreads in bax.h5 or bas.h5
    format.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ReadSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="ReferenceSet">
  <xs:annotation>
    <xs:documentation>DataSets of reference sequences. Replaces
    the reference.info.xml.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="ContigSetType"/>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
<xs:element name="GmapReferenceSet">
  <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:documentation>DataSets of reference sequences, with GMAP
↪indices.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ContigSetType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="SubreadSet">
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="SubreadSetType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="Subsets">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Subset" type="SubsetType" maxOccurs=
↪"unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="AlignmentSetMetadataType">
        <xs:complexContent>
            <xs:extension base="DataSetMetadataType">
                <xs:sequence>
                    <xs:element name="Aligner" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AlignmentSetType">
        <xs:annotation>
            <xs:documentation>Type for DataSets consisting of aligned
↪subreads and CCS reads.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:extension base="DataSetType">
                <xs:sequence>
                    <xs:element name="DataSetMetadata" type=
↪"AlignmentSetMetadataType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="BarcodeSetMetadataType">
        <xs:complexContent>
            <xs:extension base="DataSetMetadataType">
                <xs:sequence>
                    <xs:element name="BarcodeConstruction" type=
↪"xs:string"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="BarcodeSetType">
  <xs:annotation>
    <xs:documentation>Type for the Barcode DataSet.</
↪xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataSetType">
      <xs:sequence>
        <xs:element name="DataSetMetadata" type=
↪"BarcodeSetMetadataType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ContigSetMetadataType">
  <xs:complexContent>
    <xs:extension base="DataSetMetadataType">
      <xs:sequence>
        <xs:element name="Organism" type="xs:string"
↪minOccurs="0"/>
        <xs:element name="Ploidy" type="xs:string"
↪minOccurs="0"/>
        <xs:element ref="Contigs"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ContigSetType">
  <xs:annotation>
    <xs:documentation>Type for a Contig DataSet.</
↪xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataSetType">
      <xs:sequence>
        <xs:element name="DataSetMetadata" type=
↪"ContigSetMetadataType"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="DataSetMetadataType">
  <xs:annotation>
    <xs:documentation>Extend this type to provide DataSetMetadata
↪element in each DataSet.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="TotalLength" type="xs:long"/>
    <xs:element name="NumRecords" type="xs:int"/>
    <xs:element name="Provenance" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name=
↪"CommonServicesInstanceId" type="xs:IDREF" minOccurs="0"/>
          <xs:element name="CreatorUserId" type=
↪"xs:IDREF" minOccurs="0"/>
          <xs:element name="ParentJobId" type=
↪"xs:IDREF" minOccurs="0"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="ParentTool" type=
↪ "pbbase:BaseEntityType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="CreatedBy" use="required">
        <xs:simpleType>
            <xs:restriction base=
↪ "xs:string">
                <xs:enumeration value=
↪ "Instrument"/>
                <xs:enumeration value=
↪ "User"/>
                <xs:enumeration value=
↪ "AnalysisJob"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="DataSetRootType">
    <xs:choice>
        <xs:element ref="AlignmentSet" minOccurs="0"/>
        <xs:element ref="BarcodeSet" minOccurs="0"/>
        <xs:element ref="ConsensusAlignmentSet" minOccurs="0"/>
        <xs:element ref="ConsensusReadSet" minOccurs="0"/>
        <xs:element ref="ContigSet" minOccurs="0"/>
        <xs:element ref="HdfSubreadSet" minOccurs="0"/>
        <xs:element ref="ReferenceSet" minOccurs="0"/>
        <xs:element ref="SubreadSet" minOccurs="0"/>
    </xs:choice>
</xs:complexType>
<xs:complexType name="DataSetType">
    <xs:complexContent>
        <xs:extension base="pbbase:StrictEntityType">
            <xs:sequence>
                <xs:element ref="pbbase:ExternalResources"/>
                <xs:element name="Filters" minOccurs="0">
                    <xs:annotation>
                        <xs:documentation>The set of
↪ filters defined here apply to the resident data set. Should DataSet subsets be
↪ created out of this parent DataSet, each sub-DataSet may contain its own filters.</
↪ xs:documentation>
                    </xs:annotation>
                </xs:complexType>
                <xs:sequence>
                    <xs:element name=
↪ "Filter" type="pbbase:FilterType" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="DataSets" minOccurs="0">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref=
↪ "DataSet" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>

```

(continues on next page)

(continued from previous page)

```

        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ReadSetMetadataType">
  <xs:complexContent>
    <xs:extension base="DataSetMetadataType">
      <xs:sequence>
        <xs:element ref="pbmeta:Collections"
↳minOccurs="0"/>
        <xs:element ref="pbsample:BioSamples"
↳minOccurs="0"/>
        <xs:element name="SummaryStats" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name=
↳"AdapterDimerFraction" type="xs:float"/>
              <xs:element name=
↳"ShortInsertFraction" type="xs:float"/>
              <xs:element name=
↳"NumSequencingZmws" type="xs:int"/>
              <xs:element name=
↳"ProdDist" type="pbbase:StatsDiscreteDistType"/>
              <xs:element name=
↳"ReadTypeDist" type="pbbase:StatsDiscreteDistType"/>
              <xs:element name=
↳"ReadLenDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"ReadQualDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"ControlReadLenDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"ControlReadQualDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"MedianInsertDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"InsertReadLenDist" type="pbbase:StatsContinuousDistType"/>
              <xs:element name=
↳"InsertReadQualDist" type="pbbase:StatsContinuousDistType"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="ReadSetType">
  <xs:annotation>
    <xs:documentation>Type for DataSets consisting of unaligned
↳subreads and CCS reads DataSets</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="DataSetType">
      <xs:sequence>
        <xs:element name="DataSetMetadata" type=
↳"ReadSetMetadataType" minOccurs="0"/>

```

(continues on next page)

(continued from previous page)

```

        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubreadSetMetadataType">
    <xs:complexContent>
      <xs:extension base="DataSetMetadataType">
        <xs:sequence>
          <xs:element name="AverageSubreadLength" type=
↪ "xs:int"/>
          <xs:element name="AverageSubreadQuality" type=
↪ "xs:float"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubreadSetType">
    <xs:complexContent>
      <xs:extension base="ReadSetType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="SubsetType">
    <xs:complexContent>
      <xs:extension base="pbbase:StrictEntityType">
        <xs:sequence>
          <xs:element name="Filters" minOccurs="0">
            <xs:annotation>
              <xs:documentation>The set of
↪ filters defined here apply to the resident data set. Should Data_
↪ set subsets be created out of this parent DataSet, each sub-DataSet may contain its own filters.</
↪ xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name=
↪ "Filter" type="pbbase:FilterType" maxOccurs="unbounded"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element ref="pbbase:DataPointers"
↪ minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>

```

9.6 PacBioDeclData

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="UOM" type="xs:string"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="Name" type="xs:anyURI"/>
<xs:element name="Scope" type="xs:string"/>
<xs:element name="Value" type="xs:string"/>
<xs:element name="Version" type="T_Version"/>
<xs:element name="Category" type="xs:string"/>
<xs:element name="DeclData" type="T_DeclData"/>
<xs:element name="Specifier" type="T_Specifier"/>
<xs:element name="ValueType" type="xs:string"/>
<xs:element name="DeclDataSet" type="T_DeclDataSet"/>
<xs:element name="Description" type="xs:string"/>
<xs:element name="ComplexValue" type="xs:string"/>
<xs:element name="DefaultValue" type="xs:string"/>
<xs:element name="Classification" type="xs:string"/>
<xs:complexType name="T_DeclData">
  <xs:sequence>
    <xs:element ref="Classification"/>
    <xs:element ref="Name"/>
    <xs:element ref="Description"/>
    <xs:element ref="Category"/>
    <xs:element ref="DefaultValue"/>
    <xs:element ref="Value"/>
    <xs:element ref="ValueType"/>
    <xs:element ref="UOM"/>
    <xs:element ref="Scope"/>
    <xs:element ref="Version"/>
    <xs:element ref="Specifier"/>
    <xs:element ref="ComplexValue"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_Specifier">
</xs:complexType>
<xs:complexType name="T_DeclDataSet">
  <xs:sequence>
    <xs:element ref="DeclData" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:simpleType name="T_Version">
  <xs:restriction base="xs:decimal">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

9.7 PacBioPartNumbers

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioPartNumbers.xsd" xmlns:xs=
"http://www.w3.org/2001/XMLSchema" xmlns:xi="http://www.w3.org/2003/XInclude"
xmlns:xml="http://www.w3.org/XML/1998/namespace" xmlns:pbbase="http://
pacificbiosciences.com/PacBioBaseDataModel.xsd" xmlns:pbdm="http://
pacificbiosciences.com/PacBioDataModel.xsd" xmlns:pbrk="http://pacificbiosciences.com/PacBioReagentKit.xsd" targetNamespace="http://pacificbiosciences.com/
PacBioPartNumbers.xsd" elementFormDefault="qualified" id="PacBioPartNumbers">

```

(continues on next page)

(continued from previous page)

```

<xs:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation=
↪ "http://www.w3.org/2001/03/xml.xsd"/>
<xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↪ schemaLocation="PacBioBaseDataModel.xsd"/>
<xs:import namespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
↪ schemaLocation="PacBioReagentKit.xsd"/>
<xs:element name="PacBioPartNumbers" type="PacBioPartNumbersType">
  <xs:annotation>
    <xs:documentation>The root element of the Part Numbers </
↪ xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="SequencingKit" type="pbrk:SupplyKitSequencing"/>
  <xs:element name="BindingKit" type="pbbase:SupplyKitBinding"/>
  <xs:element name="TemplatePrepKit" type="pbbase:SupplyKitTemplate"/>
  <xs:element name="ControlKit" type="pbbase:SupplyKitControl"/>
  <xs:element name="CellPackKit" type="pbbase:SupplyKitCellPack"/>
  <xs:element name="OtherKit">
    <xs:complexType>
      <xs:complexContent>
        <xs:extension base="pbbase:PartNumberType">
          <xs:attribute name="MaxCollections" type=
↪ "xs:int"/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="PartTypes">
    <xs:restriction base="xs:string">
      <xs:enumeration value="WFA"/>
      <xs:enumeration value="BDK"/>
      <xs:enumeration value="TPK"/>
      <xs:enumeration value="SQK"/>
      <xs:enumeration value="CPK"/>
      <xs:enumeration value="OSE"/>
      <xs:enumeration value="CMO"/>
      <xs:enumeration value="Other"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="PacBioPartNumbersType">
    <xs:sequence>
      <xs:element ref="pbbase:KeyValueMap"/>
      <xs:element name="Automations" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Automation" type=
↪ "pbbase:AutomationType" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="BindingKits" minOccurs="0">
        <xs:annotation>
          <xs:documentation>List the binding kit part
↪ numbers. A list of incompatible part numbers and automations is available to
↪ specify in the PartNumber subtype.</xs:documentation>
        </xs:annotation>
      </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
          <xs:element ref="BindingKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="TemplatePrepKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the sample prep kit
↪part numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="TemplatePrepKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="SequencingKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the sequencing kit
↪part numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="SequencingKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ControlKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the DNA control
↪complex part numbers. A list of incompatible part numbers and automations is
↪available to specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="ControlKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="CellPackKits" minOccurs="0">
      <xs:annotation>
        <xs:documentation>List the cell tray part
↪numbers. A list of incompatible part numbers and automations is available to
↪specify in the PartNumber subtype.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="CellPackKit"
↪maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:element>
        <xs:element name="OtherKits" minOccurs="0">
            <xs:annotation>
                <xs:documentation>A placeholder for
↳ miscellaneous parts, such as OS Enzyme tubes</xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="OtherKit" maxOccurs=
↳ "unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="IncompatibleParts" minOccurs="0">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="IncompatiblePart"
↳ type="pbbase:IncompatiblePairType" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:schema>

```

9.8 PacBioPrimaryMetrics

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!--
*****
**
**   File Name: PipeStats.xsd
**           XML schema defining pipeline statistics
**
**   Programmer: Aseneth Lopez, Phil McClueg
**
**   $Id$
**
*****
**   History:
**
**       2010/03/11 pmclurg - created to correspond to new Functional Specs
**
*****
-->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioPipelineStats.xsd" xmlns:xs=
↳ "http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↳ PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
↳ PacBioPipelineStats.xsd" elementFormDefault="qualified" id="PacBioPipelineStats">
    <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
↳ schemaLocation="PacBioBaseDataModel.xsd"/>
<!--

```

(continues on next page)

(continued from previous page)

```

Movie Metrics and Classification

-->
    <!--
MovieContext
-->
    <xs:element name="MovieName" type="xs:string"/>
    <!--
MovieLength
-->
    <xs:element name="MovieLength" type="xs:double"/>
    <!--
NumDroppedFrames
-->
    <xs:element name="NumFramesDropped" type="xs:double"/>
    <!--
NumSequencingZmws
-->
    <xs:element name="NumSequencingZmws" type="xs:double"/>
    <!--
TraceFileSize
-->
    <xs:element name="TraceFileSize" type="xs:long"/>
    <!--
PulseFileSize
-->
    <xs:element name="PulseFileSize" type="xs:long"/>
    <!--
BaseFileSize
-->
    <xs:element name="BaseFileSize" type="xs:long"/>
    <!--
AdapterDimerFraction
-->
    <xs:element name="AdapterDimerFraction" type="xs:double"/>
    <!--
ShortInsertFraction
-->
    <xs:element name="ShortInsertFraction" type="xs:double"/>
    <!--
FractionFailedZmwClippedLow
-->
    <xs:element name="FailedZmwClippedLowFraction" type="xs:double"/>
    <!--
FractionFailedZmwClippedHigh
-->
    <xs:element name="FailedZmwClippedHighFraction" type="xs:double"/>
    <!--
TotalBaseFraction
-->
    <xs:element name="TotalBaseFractionValue" type="xs:double"/>
    <xs:complexType name="TotalBaseFraction">
        <xs:sequence>
            <xs:element ref="TotalBaseFractionValue"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string" use="required"/>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

    <xs:element name="TotalBaseFractionPerChannel" type="TotalBaseFraction"/>
    <!--
PkMidCV
-->
    <xs:element name="PkMidCVValue" type="xs:double"/>
    <xs:complexType name="PkMidCV">
        <xs:sequence>
            <xs:element ref="PkMidCVValue"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:element name="PkMidCVPerChannel" type="PkMidCV"/>
    <!--
Total number of counts in the sample
-->
    <xs:element name="SampleSize" type="xs:int">
        <xs:annotation>
            <xs:documentation>Number of data values in the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Mean of the sample
-->
    <xs:element name="SampleMean" type="xs:double">
        <xs:annotation>
            <xs:documentation>Mean of the sample</xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Median of the sample
-->
    <xs:element name="SampleMed" type="xs:double">
        <xs:annotation>
            <xs:documentation>Median of the sample</xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Standard Deviation of the sample
-->
    <xs:element name="SampleStd" type="xs:double">
        <xs:annotation>
            <xs:documentation>StandardDeviation of the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
95th Percentile of the sample
-->
    <xs:element name="Sample95thPct" type="xs:double">
        <xs:annotation>
            <xs:documentation>95th Percentile of the sample</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
Number of Bins in Histogram

```

(continues on next page)

(continued from previous page)

```

-->
    <xs:element name="NumBins" type="xs:int">
        <xs:annotation>
            <xs:documentation>Number of Bins in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Bin Counts in Histogram
-->
    <xs:element name="BinCount" type="xs:int">
        <xs:annotation>
            <xs:documentation>Bin Counts in Histogram</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Bin Width in Histogram
-->
    <xs:element name="BinWidth" type="xs:double">
        <xs:annotation>
            <xs:documentation>Bin Width in Histogram</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Histogram outlier minimum value
-->
    <xs:element name="MinOutlierValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Minimum Outlier Value in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram minimum bin value (less outliers)
-->
    <xs:element name="MinBinValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Minimum Value (less outliers) in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram maximum bin value (less outliers)
-->
    <xs:element name="MaxBinValue" type="xs:double">
        <xs:annotation>
            <xs:documentation>Maximum Value (less outliers) in Histogram</
↪xs:documentation>
            </xs:annotation>
        </xs:element>
    <!--
    Histogram outlier maximum value
-->
    <xs:element name="MaxOutlierValue" type="xs:double">
        <xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:documentation>Maximum Value in Histogram</
->xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    "Human-Readable" description of movie metric
    -->
    <xs:element name="MetricDescription" type="xs:string">
        <xs:annotation>
            <xs:documentation>Description of metric</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Bin Labels for discrete distributions
    -->
    <xs:element name="BinLabel" type="xs:string">
        <xs:annotation>
            <xs:documentation>Bin Label for Discrete Distribution</
->xs:documentation>
        </xs:annotation>
    </xs:element>
    <!--
    Continuous distribution class
    <xs:complexType name="ContinuousDist">
        <xs:sequence>
            <xs:element ref="SampleSize"/>
            <xs:element ref="SampleMean"/>
            <xs:element ref="SampleMed"/>
            <xs:element ref="SampleStd"/>
            <xs:element ref="Sample95thPct"/>
            <xs:element ref="NumBins"/>
            <xs:element ref="BinCount" minOccurs="1" maxOccurs="unbounded
->"/>
            <xs:element ref="BinWidth"/>
            <xs:element ref="MinOutlierValue"/>
            <xs:element ref="MinBinValue"/>
            <xs:element ref="MaxBinValue"/>
            <xs:element ref="MaxOutlierValue"/>
            <xs:element ref="MetricDescription"/>
        </xs:sequence>
        <xs:attribute name="Channel" type="xs:string"/>
    </xs:complexType> -->
    <!--
    Discrete distribution class
    <xs:complexType name="DiscreteDist">
        <xs:sequence>
            <xs:element ref="NumBins"/>
            <xs:element ref="BinCount" minOccurs="1" maxOccurs="unbounded
->"/>
            <xs:element ref="MetricDescription"/>
            <xs:element ref="BinLabel" minOccurs="1" maxOccurs="unbounded
->"/>
        </xs:sequence>
    </xs:complexType> -->
    <!--
    Time series (for time-dependent metrics)
    <xs:complexType name="TimeSeries">

```

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
            <xs:element name="TimeUnits" type="xs:string"/>
            <xs:element name="ValueUnits" type="xs:string"/>
            <xs:element name="StartTime" type="xs:float"/>
            <xs:element name="MeasInterval" type="xs:float"/>
            <xs:element name="Val" type="xs:float" minOccurs="0"
↪maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType> -->
    <!--
Specific Distributions
-->
    <!--
Control Read Length Distribution
-->
        <xs:element name="ControlReadLenDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Control Read Quality Distribution
-->
        <xs:element name="ControlReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Baseline Level Distribution - all holes
-->
        <xs:element name="BaselineLevelDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Baseline Standard Deviation Distribution - all holes
-->
        <xs:element name="BaselineStdDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Movie Read Quality Distribution - all holes
-->
        <xs:element name="MovieReadQualDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Productivity Distribution - all holes
-->
        <xs:element name="ProdDist" type="pbbase:StatsDiscreteDistType"/>
    <!--
ReadType Distribution
-->
        <xs:element name="ReadTypeDist" type="pbbase:StatsDiscreteDistType"/>
    <!--
Pulse Rate Distribution - productive holes
-->
        <xs:element name="PulseRateDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Mean Pulse Width Distribution - productive holes
-->
        <xs:element name="PulseWidthDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Base Rate (global) Distribution - productive holes, HQ regions
-->
        <xs:element name="BaseRateDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Mean Base Width Distribution - productive holes, HQ regions
-->
        <xs:element name="BaseWidthDist" type="pbbase:StatsContinuousDistType"/>
    <!--

```

(continues on next page)

(continued from previous page)

```

Mean Base IPD Distribution - productive holes, HQ regions
-->
  <xs:element name="BaseIpdDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Local Base Rate Distribution - productive holes, HQ regions.
I.e., an estimate of the polymerase rate, excluding pauses.
-->
  <xs:element name="LocalBaseRateDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Number of unfiltered basecalls Distribution - productive holes
-->
  <xs:element name="NumUnfilteredBasecallsDist" type=
→ "pbbase:StatsContinuousDistType"/>
  <!--
Polymerase Read Length Distribution - productive holes
-->
  <xs:element name="ReadLenDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Polymerase Read Quality Distribution - productive holes
-->
  <xs:element name="ReadQualDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Distribution of the ratio (Polymerase Read Len)/(Num Unfiltered Basecalls)
-->
  <xs:element name="HqBaseFractionDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Polymerase Read-Mean Base Quality Value Distribution - productive holes
-->
  <xs:element name="RmBasQvDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Best Estimate (Insert) Read Length Distribution
-->
  <xs:element name="InsertReadLenDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Best Estimate (Insert) Read Quality (Predicted Accuracy) Distribution
-->
  <xs:element name="InsertReadQualDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Local Yield Distribution - % of productive holes locally
-->
  <xs:element name="LocalYieldDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Local Snr Distribution - snr per channel locally
-->
  <xs:element name="LocalSnrDist" type="pbbase:StatsContinuousDistType"/>
  <!--
Trace Clipped Fraction Distribution - Total % traces clipped per channel
-->
  <xs:element name="TraceClippedFractionDist" type=
→ "pbbase:StatsContinuousDistType"/>
  <!--
Trace Low Clipped Fraction Distribution - % traces clipped low per channel
-->
  <xs:element name="TraceLowClippedFractionDist" type=
→ "pbbase:StatsContinuousDistType"/>
  <!--
Trace High Clipped Fraction Distribution - % traces clipped high per channel

```

(continues on next page)

(continued from previous page)

```

-->
  <xs:element name="TraceHighClippedFractionDist" type=
↪ "pbbase:StatsContinuousDistType"/>
  <!--
Pausiness Distribution - % ipds above a threshold
-->
    <xs:element name="PausinessDist" type="pbbase:StatsContinuousDistType"/>
    <!--
Adapter Dimer Distribution - % distribution of mean insert length
-->
      <xs:element name="MedianInsertDist" type="pbbase:StatsContinuousDistType"/>
      <!--
PkMid CV Distribution - % distribution of coefficient of variation of PkMid
-->
        <xs:element name="PkMidCVDist" type="pbbase:StatsContinuousDistType"/>
        <!--
SNR Distribution - productive holes
-->
          <xs:element name="SnrDist" type="pbbase:StatsContinuousDistType"/>
          <!--
HQ Region SNR Distribution - robust estimate of SNR distribution over productive_
↪ holes
-->
            <xs:element name="HqRegionSnrDist" type="pbbase:StatsContinuousDistType"/>
            <!--
HqBasPkMid Distribution - productive holes
-->
              <xs:element name="HqBasPkMidDist" type="pbbase:StatsContinuousDistType"/>
              <!--
BaselineLevelSequencing Distribution - productive holes
-->
                <xs:element name="BaselineLevelSequencingDist" type=
↪ "pbbase:StatsContinuousDistType"/>
                <!--
BaselineLevelAntihole Distribution - productive holes
-->
                  <xs:element name="BaselineLevelAntiholeDist" type=
↪ "pbbase:StatsContinuousDistType"/>
                  <!--
BaselineLevelAntimirror Distribution - productive holes
-->
                    <xs:element name="BaselineLevelAntimirrorDist" type=
↪ "pbbase:StatsContinuousDistType"/>
                    <!--
BaselineLevelFiducial Distribution - productive holes
-->
                      <xs:element name="BaselineLevelFiducialDist" type=
↪ "pbbase:StatsContinuousDistType"/>
                      <!--
Subsystem Failure Metrics, Added in Software v1.3.1
-->
                        <xs:element name="MaxPauseFractionVsT" type="xs:double"/>
                        <xs:element name="TMaxPauseFraction" type="xs:int"/>
                        <xs:element name="MaxSlopePauseFractionVsT" type="xs:double"/>
                        <xs:element name="TMaxSlopePauseFraction" type="xs:int"/>
                        <xs:element name="MaxBaseRateRatioVsT" type="xs:double"/>
                        <xs:element name="TMaxBaseRateRatio" type="xs:int"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element name="MaxSlopeBaseRateRatioVsT" type="xs:double"/>
<xs:element name="TMaxSlopeBaseRateRatio" type="xs:int"/>
<xs:element name="SgnMaxSlopeBaseRateRatio" type="xs:int"/>
<xs:element name="BaseRateChngStrtToEnd" type="xs:double"/>
<xs:element name="YieldCvOverRegions" type="xs:double"/>
<xs:element name="YieldChngCntrToEdge" type="xs:double"/>
<xs:element name="SnrRatioEdgeToCntr_0" type="xs:double"/>
<xs:element name="SnrRatioEdgeToCntr_2" type="xs:double"/>
<xs:element name="PauseFractionVsT" type="pbbase:StatsTimeSeriesType"/>
<xs:element name="BaseRateRatioVsT" type="pbbase:StatsTimeSeriesType"/>
<!--
Other v1.3.1 Metric Additions
-->
<xs:element name="IsReadsFraction" type="xs:double"/>
<!--
v2.0 Additions - spectral diagnostics
-->
<xs:element name="SpectralDiagRRDist" type="pbbase:StatsContinuousDistType"/>
<!--
PipeStats class for PipelineStats XML
-->
<xs:element name="PipeStats">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="MovieName" minOccurs="1" maxOccurs="1"
↪"/>
      <xs:element ref="MovieLength" minOccurs="1" maxOccurs="
↪"1"/>
      <xs:element ref="NumFramesDropped" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="NumSequencingZmws" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="TraceFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="PulseFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="BaseFileSize" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="AdapterDimerFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="ShortInsertFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="IsReadsFraction" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="FailedZmwClippedLowFraction"
↪minOccurs="1" maxOccurs="1"/>
      <xs:element ref="FailedZmwClippedHighFraction"
↪minOccurs="1" maxOccurs="1"/>
      <xs:element ref="ControlReadLenDist" minOccurs="0"
↪maxOccurs="1"/>
      <xs:element ref="ControlReadQualDist" minOccurs="0"
↪maxOccurs="1"/>
      <xs:element ref="ProdDist" minOccurs="1" maxOccurs="1"
↪"/>
      <xs:element ref="ReadTypeDist" minOccurs="1"
↪maxOccurs="1"/>
      <xs:element ref="TotalBaseFractionPerChannel"
↪minOccurs="4" maxOccurs="4"/>

```

(continues on next page)

(continued from previous page)

```

↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪minOccurs="1" maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪"1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪maxOccurs="4"/>
↪"4" maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪maxOccurs="1"/>
↪maxOccurs="1"/>
↪>
↪maxOccurs="4"/>
↪maxOccurs="4"/>
↪minOccurs="4" maxOccurs="4"/>
↪"4" maxOccurs="4"/>

```

```

<xs:element ref="PkMidCVPerChannel" minOccurs="4"
<xs:element ref="BaselineLevelDist" minOccurs="4"
<xs:element ref="BaselineStdDist" minOccurs="4"
<xs:element ref="MovieReadQualDist" minOccurs="1"
<xs:element ref="PulseRateDist" minOccurs="1"
<xs:element ref="PulseWidthDist" minOccurs="1"
<xs:element ref="BaseRateDist" minOccurs="1"
<xs:element ref="BaseWidthDist" minOccurs="1"
<xs:element ref="BaseIpdDist" minOccurs="1" maxOccurs=
<xs:element ref="LocalBaseRateDist" minOccurs="1"
<xs:element ref="NumUnfilteredBasecallsDist"
<xs:element ref="ReadLenDist" minOccurs="1" maxOccurs=
<xs:element ref="ReadQualDist" minOccurs="1"
<xs:element ref="HqBaseFractionDist" minOccurs="1"
<xs:element ref="RmBasQvDist" minOccurs="1" maxOccurs=
<xs:element ref="InsertReadLenDist" minOccurs="0"
<xs:element ref="InsertReadQualDist" minOccurs="0"
<xs:element ref="LocalYieldDist" minOccurs="1"
<xs:element ref="LocalSnrDist" minOccurs="4"
<xs:element ref="TraceClippedFractionDist" minOccurs=
<xs:element ref="TraceLowClippedFractionDist"
<xs:element ref="TraceHighClippedFractionDist"
<xs:element ref="PausinessDist" minOccurs="1"
<xs:element ref="MedianInsertDist" minOccurs="1"
<xs:element ref="SnrDist" minOccurs="4" maxOccurs="4"/
<xs:element ref="HqRegionSnrDist" minOccurs="4"
<xs:element ref="HqBasPkMidDist" minOccurs="4"
<xs:element ref="BaselineLevelSequencingDist"
<xs:element ref="BaselineLevelAntiholeDist" minOccurs=

```

(continues on next page)

```
<minOccurs="4" maxOccurs="4"/>  
    <"4" maxOccurs="4"/>  
    <maxOccurs="4"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <"1" maxOccurs="1"/>  
    <" " maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <"1" maxOccurs="1"/>  
    <" " maxOccurs="1"/>  
    <"1" maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
    <maxOccurs="1"/>  
  
        </xs:sequence>  
        <xs:attribute name="Version" type="xs:string" use="optional">  
            <xs:annotation>  
                <xs:documentation>An optional identifier  
denoting the revision of this particular entity</xs:documentation>  
            </xs:annotation>  
        </xs:attribute>  
    </xs:complexType>  
</xs:element>  
</xs:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 4 sp1 (x64) (http://www.altova.com) by
refavad@pacificbiosciences.com (Pacific Biosciences) -->
```

Chapter 9. Data Model XSD

(continued from previous page)

```

<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns="http://pacificbiosciences.com/PacBioReagentKit.xsd" xmlns:xs="http://
  ↪www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
  ↪PacBioBaseDataModel.xsd" xmlns:pbdm="http://pacificbiosciences.com/PacBioDataModel.
  ↪xsd" targetNamespace="http://pacificbiosciences.com/PacBioReagentKit.xsd"
  ↪elementFormDefault="qualified" id="PacBioReagentKit">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
  ↪schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:element name="PacBioReagentKit">
    <xs:annotation>
      <xs:documentation>The root element of the reagent kit
  ↪standalone file</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="ReagentKit"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Reagent" type="ReagentType"/>
  <xs:element name="ReagentKit" type="ReagentKitType"/>
  <xs:element name="ReagentTube" type="ReagentTubeType"/>
  <xs:element name="ReagentPlateRow" type="ReagentPlateRowType"/>
  <xs:complexType name="SupplyKitSequencing">
    <xs:annotation>
      <xs:documentation>A more specific template kit representation
  ↪(includes SupplyKit fields). </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="pbbase:PartNumberType">
        <xs:sequence>
          <xs:element name="ReagentAutomationName" type=
  ↪"xs:string" minOccurs="0">
            <xs:annotation>
              <xs:documentation>The reagent-
  ↪mixing protocol used. </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="ReagentTubes" type=
  ↪"SupplyKitSequencing" minOccurs="0" maxOccurs="2">
            <xs:annotation>
              <xs:documentation>Tubes
  ↪associated with the reagent kit - can have up to two; don't forget to set the
  ↪location, 0 or 1</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="SequencingChemistry" type=
  ↪"pbbase:SequencingChemistry" minOccurs="0"/>
          <xs:element name="SequencingKitDefinition"
  ↪type="ReagentKitType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="Location" type="xs:int" default="0
  ↪">
          <xs:annotation>
            <xs:documentation>The location of the
  ↪supply kit - for a reagent plate, it could be 0 or 1, and for a tube it could be 0
  ↪or 1</xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        </xs:attribute>
        <xs:attribute name="MaxCollections" type="xs:int"
↪ default="8">

        <xs:annotation>
            <xs:documentation>The number of
↪ collections this supply kit is capable of</xs:documentation>
        </xs:annotation>
        </xs:attribute>
    </xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="ReagentKey" type="ReagentKey" use=
↪ "required"/>
            <xs:attribute name="PlateColumn" type="xs:string" use=
↪ "required"/>
            <xs:attribute name="Volume" type="xs:int" use=
↪ "required"/>
            <xs:attribute name="DeadVolume" type="xs:int" use=
↪ "required"/>
            <xs:attribute name="ActiveInHour" type="xs:int" use=
↪ "required"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentKitType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:sequence>
                <xs:element name="Reagents">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪ "Reagent" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ReagentTubes">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪ "ReagentTube" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ReagentPlateRows">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref=
↪ "ReagentPlateRow" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:element ref="pbbase:ChemistryConfig"/>

```

(continues on next page)

(continued from previous page)

```

<xs:element ref="pbbase:Defaults" minOccurs="0"/>
↪ "/>
<xs:element name="Automations" nillable="true"
↪ " minOccurs="0">
    <xs:annotation>
        <xs:documentation>Automations_
↪ that are deemed compatible with this kit. Parameters specified within an_
↪ automation will override a parameter with the same name and data type specified in_
↪ the above Defaults section</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name=
↪ "Automation" type="pbbase:AutomationType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="ProductCode" type="xs:string"/>
<xs:attribute name="PlateType" type="xs:string"/>
<xs:attribute name="ActiveInHour" type="xs:int"/>
<xs:attribute name="BasesPerSecond" type="xs:decimal"/
↪ >
    <xs:attribute name="AcquisitionCount" type="xs:int"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentTubeType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="ProductCode" type="xs:string" use=
↪ "required"/>
            <xs:attribute name="ReagentKey" type="ReagentKey" use=
↪ "required"/>
            <xs:attribute name="Volume" type="xs:short" use=
↪ "required"/>
            <xs:attribute name="UsageVolume" type="xs:double" use=
↪ "required"/>
            <xs:attribute name="UsageKey" type="xs:string" use=
↪ "required"/>
            <xs:attribute name="DeadVolume" type="xs:short" use=
↪ "required"/>
            <xs:attribute name="ActiveInHour" type="xs:int" use=
↪ "required"/>
            <xs:attribute name="TubeWellType" type="TubeSize" use=
↪ "required"/>
            <xs:attribute name="ReagentTubeType" type=
↪ "TubeLocation" use="required"/>
            <xs:attribute name="InitialUse" type="xs:dateTime"
↪ use="optional"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
<xs:complexType name="ReagentPlateRowType">
    <xs:complexContent>
        <xs:extension base="pbbase:BaseEntityType">
            <xs:attribute name="PlateRow" type="xs:string" use=
↪ "required"/>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name="InitialUse" type="xs:dateTime"
↪use="optional"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:simpleType name="TubeLocation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ReagentTube0"/>
    <xs:enumeration value="ReagentTube1"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="TubeSize">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DeepTube"/>
    <xs:enumeration value="ShallowTube"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="ReagentKey">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Base"/>
    <xs:enumeration value="DTT"/>
    <xs:enumeration value="DilBuffer"/>
    <xs:enumeration value="MineralOil"/>
    <xs:enumeration value="MIXED_DilBuffer"/>
    <xs:enumeration value="MIXED_OS"/>
    <xs:enumeration value="OSbuffer"/>
    <xs:enumeration value="OSenzyme"/>
    <xs:enumeration value="PhospholinkedNT"/>
    <xs:enumeration value="SABuffer"/>
    <xs:enumeration value="Spike"/>
    <xs:enumeration value="Streptavidin"/>
    <xs:enumeration value="SubstrateOS"/>
    <xs:enumeration value="TSQ"/>
    <xs:enumeration value="WashBuffer"/>
    <xs:enumeration value="WettingBuffer"/>
    <xs:enumeration value="PCA"/>
    <xs:enumeration value="PCD"/>
    <xs:enumeration value="Analog"/>
    <xs:enumeration value="Sample"/>
    <xs:enumeration value="PEGDilBuffer"/>
    <xs:enumeration value="ExtraBuffer"/>
    <xs:enumeration value="PrewetBuffer"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

9.10 PacBioRightsAndRoles

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
↪efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioRightsAndRoles.xsd" xmlns:xs=
↪"http://www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
↪PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
↪PacBioRightsAndRoles.xsd" elementFormDefault="qualified" id="PacBioRightsAndRoles">
(continues on next page)

```

(continued from previous page)

```

<xs:import namespace="http://pacifichiosciences.com/PacBioBaseDataModel.xsd"
↳schemaLocation="PacBioBaseDataModel.xsd"/>
  <xs:element name="RnR">
    <xs:annotation>
      <xs:documentation>Rights and Roles root element</
↳xs:documentation>
      </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Roles" minOccurs="0"/>
        <xs:element ref="AccessRights" minOccurs="0"/>
        <xs:element ref="UserIdentities" minOccurs="0"/>
        <xs:element ref="Projects" minOccurs="0">
          <xs:annotation>
            <xs:documentation>A project is of
↳type AccessRight and can be defined the same way, defined by a URI (ResoureId
↳attribute) and access controlled by using the AccessDisabled attribute, set to
↳'false' to grant users access.

A user (with a role, which has access rights) has one more projects with which they
↳are associated. Via this association, one or more users may own a project, and one
↳or more users may be granted access to a project, simply by being associated with
↳it.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element ref="AuditableEvents" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="CFRp11Enabled" type="xs:boolean" use=
↳"optional" default="false">
        <xs:annotation>
          <xs:documentation>Define whether or not
↳21CFRp11 capabilities are turned on or off (default)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="LDAPUri" type="xs:anyURI">
        <xs:annotation>
          <xs:documentation>LDAP server URI used to
↳authenticate users. If none is specified, local user/password information is used
↳to determine access.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="UserIdentityType">
    <xs:complexContent>
      <xs:extension base="pbbase:StrictEntityType">
        <xs:sequence>
          <xs:element ref="Person">
            <xs:annotation>
              <xs:documentation>An entity
↳to store persons' names and contact info, to associate to users</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element ref="UserPassword">
            <xs:annotation>
              <xs:documentation>An entity
↳storing hashed password and salt - cached here in case LDAP is not availabel to
↳authenticate, or the user does not exist in LDAP (i.e. adhoc user).</ (continues on next page)
            </xs:documentation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

```

(continued from previous page)

```

        </xs:annotation>
      </xs:element>
      <xs:element ref="ProjectReferences" minOccurs=
↪ "0">

        <xs:annotation>
          <xs:documentation>List of ↪
↪ projects that the user has access to</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="UserName" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:maxLength value="50"/>
          <xs:minLength value="1"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="TokenId" type="xs:string">
      <xs:annotation>
        <xs:documentation>A token to use for ↪
↪ authentication, instead of constantly pingping LDAP with user name/password</
↪ xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="UseLDAPUri" type="xs:boolean">
      <xs:annotation>
        <xs:documentation>If the LDAP server ↪
↪ is specified at the root element, we use the UserName to authenticate against that -
↪ i.e. no password info is necessary to be stored.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Email">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:maxLength value="96"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="MessageAddress">
      <xs:annotation>
        <xs:documentation>e.g. a phone number ↪
↪ to text a message, a twitter handle...</xs:documentation>
      </xs:annotation>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="255"/>
      </xs:restriction>
    </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="AccessGrantDate" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:dateTime">
          <xs:minInclusive value="1000-
↪ 01-01T00:00:00"/>
          <xs:maxInclusive value="9999-
↪ 12-31T23:59:59"/>

```

(continues on next page)

(continued from previous page)

```

<xs:pattern value="\p{Nd}{4}-
↪\p{Nd}{2}-\p{Nd}{2}T\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="RoleReference" type="xs:IDREF"/>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="RoleType">
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType"/>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="AccessRightType">
  <xs:complexContent>
    <xs:extension base="pbbase:StrictEntityType">
      <xs:attribute name="InternalResourceAddress" use=
↪"required">
        <xs:annotation>
          <xs:documentation>e.g. svc://admin
This should support a wildcard specification, such that an entire hierarchy can be_
↪disabled via http://*/Analysis/*</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:maxLength value="50"/>
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="ComponentName">
        <xs:annotation>
          <xs:documentation>e.g. System_
↪Administration</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:maxLength value="255"/>
            <xs:minLength value="1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="AccessDisabled" type="xs:boolean"
↪use="optional" default="false">
        <xs:annotation>
          <xs:documentation>When the object is_
↪created, by default it is to disable access to a function. This should be set to
↪'false' in order to explicitly enable a function.</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="Operation" type="xs:string">
        <xs:annotation>
          <xs:documentation>Define an optional_
↪operation to the resource identifier, e.g. GET, PUT, POST</xs:documentation>
        </xs:annotation>
      </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

<xs:attribute name="RequiresAudit" type="xs:boolean"
↪use="required"/>
<xs:attribute name="RequiresESig" type="xs:boolean"
↪use="required">
    <xs:annotation>
        <xs:documentation>An ESig may only be
↪required, if an audit is required.</xs:documentation>
    </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>
<xs:complexType name="AuditableEventType">
    <xs:complexContent>
        <xs:extension base="pbbase:StrictEntityType">
            <xs:sequence>
                <xs:element ref="EventToken">
                    <xs:annotation>
                        <xs:documentation>A handoff
↪event token to be used between system components - maybe superceded by the user's
↪tokenId</xs:documentation>
                    </xs:annotation>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="DateCreated" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:dateTime">
                        <xs:minInclusive value="1000-
↪01-01T00:00:00"/>
                        <xs:maxInclusive value="9999-
↪12-31T23:59:59"/>
                        <xs:pattern value="\p{Nd}{4}-
↪\p{Nd}{2}-\p{Nd}{2}T\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="AuditEventType" use="required">
                <xs:annotation>
                    <xs:documentation>e.g. NewUserCreated
↪</xs:documentation>
                </xs:annotation>
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="OldValue">
                <xs:simpleType>
                    <xs:restriction base="xs:base64Binary"
↪">
                    <xs:maxLength value=
↪"2147483647"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="NewValue" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:base64Binary"
↳ ">
              <xs:maxLength value=
↳ "2147483647"/>
              <xs:minLength value="1"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="Reason">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="255"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="ESig">
          <xs:simpleType>
            <xs:restriction base="xs:base64Binary"
↳ ">
              <xs:maxLength value=
↳ "2147483647"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="UserIdentityReference" type=
↳ "xs:IDREF"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ProjectReferenceType">
    <xs:attribute name="ProjectId" type="xs:IDREF" use="required">
      <xs:annotation>
        <xs:documentation>Reference to a project ID</
↳ xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="IsOwner" type="xs:boolean">
      <xs:annotation>
        <xs:documentation>Designate if the user has ownership
↳ of this project; note that multiple users may have ownership of the same project.</
↳ xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="UserIdentity" type="UserIdentityType">
    <xs:annotation>
      <xs:documentation>A user entity relating to a person
↳ (optional) and role;</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="Role">
    <xs:annotation>
      <xs:documentation>The role a user would play in the system, e.
↳ g. Admin, tech</xs:documentation>
    </xs:annotation>

```

(continues on next page)

(continued from previous page)

```

        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="RoleType">
              <xs:sequence>
                <xs:element ref="AccessRightReferences"
↳ " minOccurs="0"/>
                <xs:element ref=
↳ "UserIdentityReferences" minOccurs="0"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="AccessRight">
        <xs:annotation>
          <xs:documentation>Define the functions that a role is capable
↳ of accessing</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="AccessRightType">
              <xs:sequence>
                <xs:element ref="RoleReferences"/>
                <xs:element ref=
↳ "AuditableEventReferences" minOccurs="0"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <xs:element name="AuditableEvent" type="AuditableEventType">
        <xs:annotation>
          <xs:documentation>An auditable record is created if required
↳ for an access right</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="Project" type="AccessRightType">
        <xs:annotation>
          <xs:documentation>A project is of type AccessRight and can be
↳ defined the same way, defined by a URI (ResourceId attribute) and access controlled
↳ by using the AccessDisabled attribute, set to 'false' to grant users access.

A user (with a role, which has access rights) has one more projects with which
↳ they're associated. Via this association, one or more users may own a project, and
↳ one or more users may be granted access to a project, simply by being associated
↳ with it.
</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="EventToken">
        <xs:annotation>
          <xs:documentation>A handoff event token to be used between
↳ system components o</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:attribute name="Token">
            <xs:simpleType>

```

(continues on next page)

(continued from previous page)

```

        <xs:restriction base="xs:base64Binary">
            <xs:maxLength value="255"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>
<xs:element name="Person">
    <xs:annotation>
        <xs:documentation>a table to store persons' full names, to
→associate to users</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:attribute name="FirstName">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="MiddleName">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="LastName" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="50"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="PhoneNumber" type="xs:string"/>
        <xs:attribute name="EMail" type="xs:string"/>
        <xs:attribute name="NotifyBySMS" type="xs:boolean"/>
        <xs:attribute name="NotifyByEmail" type="xs:boolean"/>
    </xs:complexType>
</xs:element>
<xs:element name="UserPassword">
    <xs:annotation>
        <xs:documentation>a table storing hashed password and salt</
→xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:attribute name="PasswordHash" use="required">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                    <xs:minLength value="1"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="Salt" use="required">
            <xs:simpleType>

```

(continues on next page)

(continued from previous page)

```

        <xs:restriction base="xs:string">
            <xs:maxLength value="255"/>
            <xs:minLength value="1"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
<xs:attribute name="Expired" type="xs:boolean" use="required"/>
</>

    </xs:complexType>
</xs:element>
<xs:element name="Roles">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more Role elements
    </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Role" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>The role a user
would play in the system, e.g. Admin, tech, scientist, PI, etc.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="AccessRights">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more AccessRight
elements</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="AccessRight" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>Define the
functions that a role is capable of accessing
                </xs:documentation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="UserIdentities">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more UserIdentity
elements</xs:documentation>
    </xs:annotation>
    <xs:complexType>

```

The name attribute should be used to define the Access Right's name. The Component_ Name is the right's parent, in the hierarchy of functional access.

The ResourceId, in this case, e.g. svc://admin, should support a wildcard_ specification, such that an entire hierarchy can be disabled via http://*/Analysis/*

The AccessDisabled attribute may be used to allow/restrict (default) functionality.

(continues on next page)

(continued from previous page)

```

        <xs:sequence>
            <xs:element ref="UserIdentity" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="AuditableEvents">
    <xs:annotation>
        <xs:documentation>An aggregation of one or more
↪ AuditableEvent elements</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="AuditableEvent" maxOccurs="unbounded
↪ "/>

        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="Projects">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Project" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>An grouping of
↪ datasets and analysis jobs, with user access controls

The Project type extends AccessRight, allowing a project to be defined by a URI
↪ (ResoureId attribute) and access controlled by using the AccessDisabled attribute,
↪ set to 'false' to grant users access.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="AccessRightReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to AccessRight element UUIDs
↪ </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="AccessRightReference" type="xs:IDREF
↪ " maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>IDREF to an
↪ AccessRight element UUID</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="UserIdentityReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to UserIdentity element UUIDs
↪ </xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>

```

(continues on next page)

(continued from previous page)

```

        <xs:element name="UserIdentityReference" type=
↪ "xs:IDREF" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>IDREF to a
↪ UserIdentity element UUID</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="AuditableEventReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to AuditableEvent element
↪ UUIDs</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="AuditableEventReference" type=
↪ "xs:IDREF" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>IDREF to an
↪ AuditableEvent element UUID</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="RoleReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to Role element UUIDs</
↪ xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="RoleReference" type="xs:IDREF"
↪ maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>IDREF to a Role
↪ element UUID</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="ProjectReferences">
    <xs:annotation>
        <xs:documentation>List of IDREFs to DataUserGroup element
↪ UUIDs</xs:documentation>
    </xs:annotation>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="ProjectReference" type=
↪ "ProjectReferenceType" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>List of 1 or more
↪ projects that the user has access to, or ownership of.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

(continues on next page)

(continued from previous page)

```

        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

9.11 PacBioSampleInfo

```

<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- edited with XMLSpy v2015 rel. 3 (x64) (http://www.altova.com) by
efayad@pacificbiosciences.com (Pacific Biosciences) -->
<?xml-stylesheet type="application/xml" href="metadata2html.xslt"?>
<xs:schema xmlns="http://pacificbiosciences.com/PacBioSampleInfo.xsd" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:pbbase="http://pacificbiosciences.com/
PacBioBaseDataModel.xsd" targetNamespace="http://pacificbiosciences.com/
PacBioSampleInfo.xsd" elementFormDefault="qualified" id="PacBioSampleInfo">
  <xs:import namespace="http://pacificbiosciences.com/PacBioBaseDataModel.xsd"
schemaLocation="PacBioBaseDataModel.xsd"/>
  <!-- The root element for the metadata structure -->
  <xs:complexType name="BarcodedSampleType">
    <xs:annotation>
      <xs:documentation>This is a data type to hold a barcoded
biological sample, or a raw biological sample - so, barcode is optional.</
xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="BioSampleType">
        <xs:sequence>
          <xs:element name="Barcodes" minOccurs="0">
            <xs:annotation>
              <xs:documentation>A list of
biological samples associated with the biological sample</xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name=
"Barcode" type="pbbase:DNABarcode" minOccurs="0" maxOccurs="unbounded">
              </xs:sequence>
            </xs:complexType>
            <xs:annotation>
              <xs:documentation>A sequence of barcodes associated with the biological sample</
xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:extension>
  </xs:complexType>
  <xs:complexType name="BioSampleType">

```

(continues on next page)

(continued from previous page)

```

<xs:annotation>
  <xs:documentation>The actual biological sample; this could be
↳prep'd, or in original form; could be bound, or annealed...</xs:documentation>
</xs:annotation>
<xs:complexContent>
  <xs:extension base="pbbase:StrictEntityType">
    <xs:sequence>
      <xs:element ref="BioSamples" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="DateReceived" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation>Date the sample was
↳received by the lab</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Organism" type="xs:string">
      <xs:annotation>
        <xs:documentation>e.g. HIV, E.coli</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Reference" type="xs:string">
      <xs:annotation>
        <xs:documentation>Name of reference,
↳or pointer to one at e.g. NCBI RefSeq</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="DNAType" type="xs:string">
      <xs:annotation>
        <xs:documentation>shotgun library,
↳amplicon, etc.</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Concentration" type="xs:float">
      <xs:annotation>
        <xs:documentation>in ng/uL, e.g. 250</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="QuantificationMethod" type=
↳"xs:string">
      <xs:annotation>
        <xs:documentation>e.g. Qubit</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="SMRTBellConcentration" type=
↳"xs:float">
      <xs:annotation>
        <xs:documentation>in ng/uL, e.g. 4.5</
↳xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="SMRTBellQuantificationMethod"
↳type="xs:string">
      <xs:annotation>
        <xs:documentation>e.g. Qubit</
↳xs:documentation>

```

(continues on next page)

(continued from previous page)

```

        </xs:annotation>
        </xs:attribute>
        <xs:attribute name="BufferName" type="xs:string">
            <xs:annotation>
                <xs:documentation>e.g. Tris HCl</
↪xs:documentation>

            </xs:annotation>
            </xs:attribute>
            <xs:attribute name="SamplePrepKit" type="xs:string">
                <xs:annotation>
                    <xs:documentation>e.g. SMRTbell_
↪Template Prep Kit</xs:documentation>

            </xs:annotation>
            </xs:attribute>
            <xs:attribute name="TargetLibrarySize" type="xs:string
↪">

                <xs:annotation>
                    <xs:documentation>2000, 10000, 20000</
↪xs:documentation>

                </xs:annotation>
                </xs:attribute>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="BioSamplePointers">
        <xs:annotation>
            <xs:documentation>Back references to other BarcodedSampleType_
↪object UniqueIds which utilize this sample</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:annotation>
                    <xs:documentation>The choice is to either_
↪point to a group of barcoded samples or a single biological sample</
↪xs:documentation>

                </xs:annotation>
                <xs:choice>
                    <xs:element ref="BarcodedSamplePointers"/>
                    <xs:element name="BioSamplePointer" type=
↪"xs:IDREF">

                        <xs:annotation>
                            <xs:documentation>Pointer to_
↪a single biological sample</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:choice>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="BarcodedSamplePointers">
        <xs:annotation>
            <xs:documentation>Back references to other BarcodedSampleType_
↪object UniqueIds which utilize this sample</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BarcodedSamplePointer" type=
↪"xs:IDREF" maxOccurs="unbounded">

```

(continues on next page)

(continued from previous page)

```

                                <xs:annotation>
                                    <xs:documentation>Pointer to a group
↳ of barcoded samples</xs:documentation>
                                </xs:annotation>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="BioSamples">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element ref="BioSample" minOccurs="0" maxOccurs=
↳ "unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="BioSample" type="BioSampleType"/>
</xs:schema>

```

9.12 PacBioSeedingData

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xs3p.xsl"?>
<!-- W3C Schema generated by XMLSpy v2014 rel. 2 (x64) (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="Dye" type="T_Dye"/>
    <xs:element name="Dyes" type="T_Dyes"/>
    <xs:element name="User" type="T_User"/>
    <xs:element name="DependOn" type="T_DependOn"/>
    <xs:element name="Protocol" type="T_Protocol"/>
    <xs:element name="PlateType" type="T_PlateType"/>
    <xs:element name="Protocols" type="T_Protocols"/>
    <xs:element name="PlateTypes" type="T_PlateTypes"/>
    <xs:element name="DefaultUsers" type="T_DefaultUsers"/>
    <xs:element name="HiveMetadata" type="T_HiveMetadata"/>
    <xs:element name="ProtocolType" type="T_ProtocolType"/>
    <xs:element name="ReminderTask" type="T_ReminderTask"/>
    <xs:element name="ProtocolTypes" type="T_ProtocolTypes"/>
    <xs:element name="ReminderTasks" type="T_ReminderTasks"/>
    <xs:element name="DependencyItem" type="T_DependencyItem"/>
    <xs:element name="ReagentKitType" type="T_ReagentKitType"/>
    <xs:element name="ReagentKitTypes" type="T_ReagentKitTypes"/>
    <xs:element name="InstDBSeedingData" type="T_InstDBSeedingData"/>
    <xs:element name="ProductPartNumber" type="T_ProductPartNumber"/>
    <xs:element name="ProtocolParameter" type="T_ProtocolParameter"/>
    <xs:element name="PipelineDependency" type="T_PipelineDependency"/>
    <xs:element name="ProductPartNumbers" type="T_ProductPartNumbers"/>
    <xs:element name="CompatibleReagentKit" type="T-CompatibleReagentKit"/>
    <xs:element name="ProtocolTypeParameter" type="T_ProtocolTypeParameter"/>
    <xs:element name="ProtocolTypeParameters" type="T_ProtocolTypeParameters"/>
    <xs:complexType name="T_Dye">
        <xs:attribute name="Base" use="required" type="AT_37"/>
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:attribute name="Nucleotide" use="required" type="AT_5"/>
        <xs:attribute name="WaveLength" use="required" type="AT_5"/>
        <xs:attribute name="DyeConcentration" use="required" type="AT_5"/>
    </xs:complexType>
    <xs:complexType name="T_Dyes">
        <xs:sequence>
            <xs:element ref="Dye" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="T_User">
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="Role" use="required" type="xs:string"/>
        <xs:attribute name="Email" use="required" type="xs:string"/>
        <xs:attribute name="LastName" use="required" type="xs:string"/>
        <xs:attribute name="Password" use="required" type="xs:string"/>
        <xs:attribute name="FirstName" use="required" type="xs:string"/>
        <xs:attribute name="Description" use="required" type="xs:string"/>
        <xs:attribute name="PhoneNumber" use="required" type="AT_5"/>
    </xs:complexType>
    <xs:complexType name="T_DependOn">
        <xs:attribute name="BindingKit" use="required" type="xs:int"/>
        <xs:attribute name="SequencingKit" use="required" type="xs:int"/>
    </xs:complexType>
    <xs:complexType name="T_Protocol">
        <xs:choice minOccurs="0">
            <xs:element ref="CompatibleReagentKit" maxOccurs="unbounded"/>
            <xs:element ref="ProtocolParameter"/>
        </xs:choice>
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
        <xs:attribute name="InternalOnly" use="required" type="xs:boolean"/>
        <xs:attribute name="ProtocolType" use="required" type="xs:string"/>
        <xs:attribute name="PythonModule" use="required" type="xs:string"/>
        <xs:attribute name="hasPreRunTasks" use="required" type="xs:boolean"/>
        <xs:attribute name="hasPostRunTasks" use="required" type="xs:boolean"/>

        <xs:attribute name="compatibleBindingKits" type="xs:string"/>
        <xs:attribute name="compatibleChipLayouts" type="xs:string"/>
        <xs:attribute name="compatibleSamplePrepKits" type="xs:string"/>
    </xs:complexType>
    <xs:complexType name="T_PlateType">
        <xs:attribute name="Name" use="required" type="xs:string"/>
        <xs:attribute name="NumOfWells" use="required" type="xs:short"/>
        <xs:attribute name="Description" use="required" type="xs:string"/>
        <xs:attribute name="MaxWellVolume" use="required" type="xs:decimal"/>
        <xs:attribute name="DeadWellVolume" use="required" type="xs:decimal"/>
    </xs:complexType>
    <xs:complexType name="T_Protocols">
        <xs:sequence>
            <xs:element ref="Protocol" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="T_PlateTypes">
        <xs:sequence>
            <xs:element ref="PlateType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

```

(continues on next page)

(continued from previous page)

```

<xs:complexType name="T_DefaultUsers">
  <xs:sequence>
    <xs:element ref="User" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_HiveMetadata">
  <xs:attribute name="HiveTableList" use="required" type="xs:string"/>
  <xs:attribute name="MetadataVersion" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolType">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ReminderTask">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="Interval" use="required" type="xs:string"/>
  <xs:attribute name="StartTime" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
  <xs:attribute name="DelegateName" use="required" type="xs:anyURI"/>
  <xs:attribute name="IsVisibleInUI" use="required" type="xs:boolean"/>
  <xs:attribute name="ReminderOffset" use="required" type="AT_11"/>
  <xs:attribute name="IsPreventsRunStart" use="required" type=
↪ "xs:boolean"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypes">
  <xs:sequence>
    <xs:element ref="ProtocolType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_ReminderTasks">
  <xs:sequence>
    <xs:element ref="ReminderTask" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_DependencyItem">
  <xs:sequence>
    <xs:element ref="DependOn"/>
  </xs:sequence>
  <xs:attribute name="ConfigFileName" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ReagentKitType">
  <xs:attribute name="Name" use="required" type="xs:string"/>
  <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
  <xs:attribute name="Description" use="required" type="xs:string"/>
  <xs:attribute name="InternalOnly" use="required" type="xs:boolean"/>
</xs:complexType>
<xs:complexType name="T_ReagentKitTypes">
  <xs:sequence>
    <xs:element ref="ReagentKitType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="T_InstDBSeedingData">
  <xs:sequence>
    <xs:element ref="DefaultUsers"/>
    <xs:element ref="ProtocolTypes"/>

```

(continues on next page)

(continued from previous page)

```

        <xs:element ref="ProtocolTypeParameters"/>
        <xs:element ref="PlateTypes"/>
        <xs:element ref="ReagentKitTypes"/>
        <xs:element ref="Protocols"/>
        <xs:element ref="Dyes"/>
        <xs:element ref="ProductPartNumbers"/>
        <xs:element ref="PipelineDependency"/>
        <xs:element ref="ReminderTasks"/>
        <xs:element ref="HiveMetadata"/>
    </xs:sequence>
    <xs:attribute name="Version" use="required" type="xs:decimal"/>
    <xs:attribute name="Description" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProductPartNumber">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="DyeNames" type="xs:string"/>
    <xs:attribute name="ChipLayout" type="xs:string"/>
    <xs:attribute name="IsObsolete" use="required" type="xs:boolean"/>
    <xs:attribute name="PartNumber" use="required" type="xs:int"/>
    <xs:attribute name="ControlName" type="xs:string"/>
    <xs:attribute name="Description" use="required" type="xs:string"/>
    <xs:attribute name="ProductName" use="required" type="xs:string"/>
    <xs:attribute name="PolymeraseSpeed" type="xs:decimal"/>
    <xs:attribute name="PolymeraseKinetic" type="xs:short"/>
    <xs:attribute name="ControlDisplayName" type="xs:string"/>
    <xs:attribute name="LeftPrimerSequence" type="xs:string"/>
    <xs:attribute name="LeftAdaptorSequence" type="xs:string"/>
    <xs:attribute name="RightPrimerSequence" type="xs:string"/>
    <xs:attribute name="RightAdaptorSequence" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolParameter">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="Value" use="required" type="xs:string"/>
    <xs:attribute name="ValueType" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_PipelineDependency">
    <xs:sequence>
        <xs:element ref="DependencyItem" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="T_ProductPartNumbers">
    <xs:sequence>
        <xs:element ref="ProductPartNumber" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="T_CompatibleReagentKit">
    <xs:attribute name="PPN" use="required" type="xs:int"/>
    <xs:attribute name="MixingProtocol" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypeParameter">
    <xs:attribute name="Name" use="required" type="xs:string"/>
    <xs:attribute name="MethodName" use="required" type="xs:string"/>
    <xs:attribute name="PythonModule" use="required" type="xs:string"/>
    <xs:attribute name="ProtocolTypes" use="required" type="xs:string"/>
</xs:complexType>
<xs:complexType name="T_ProtocolTypeParameters">
    <xs:sequence>

```

(continues on next page)

(continued from previous page)

```

<xs:element ref="ProtocolTypeParameter" maxOccurs="unbounded"/
</xs:sequence>
</xs:complexType>
<xs:simpleType name="AT_5">
  <xs:restriction base="xs:string">
    <xs:enumeration value=""/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="AT_11">
  <xs:restriction base="xs:byte">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="AT_37">
  <xs:restriction base="xs:string">
    <xs:enumeration value="A"/>
    <xs:enumeration value="C"/>
    <xs:enumeration value="G"/>
    <xs:enumeration value="T"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

EXHIBIT H

HIFI SEQUENCING

HiFi reads for highly accurate long-read sequencing

Connect with a PacBio scientist

TECHNOLOGIES

HiFi long-read sequencing

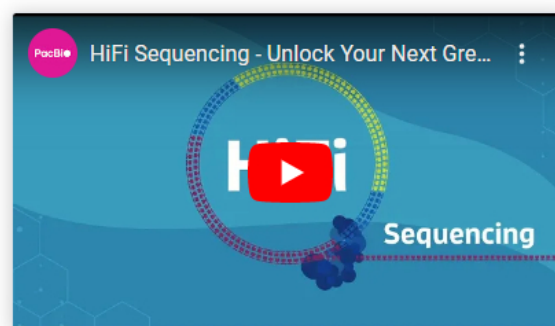
SBB short-read sequencing

Nanobind DNA extraction

HIFI SEQUENCING

UNLOCK YOUR NEXT GREAT DISCOVERY

With HiFi sequencing you get the benefits of short reads and traditional long reads in one easy-to-use technology. Watch this short video to learn how HiFi sequencing is empowering scientists to strive for new breakthroughs.

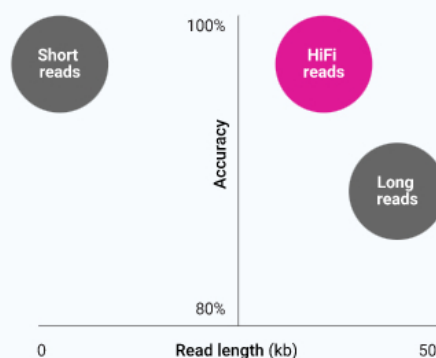


EXPLORE A NEW PARADIGM IN SEQUENCING WITH HIFI READS

Advanced scientific discoveries require sequencing data that is both accurate and complete. PacBio sequencing technology has evolved to a different type of long read, known as highly accurate long reads, or HiFi reads.

PacBio is the only sequencing technology to offer HiFi reads that provide accuracy of 99.9%, on par with short reads and Sanger sequencing. With HiFi reads you no longer have to compromise long read lengths for high accuracy sequencing to address your toughest biological questions.

How it works

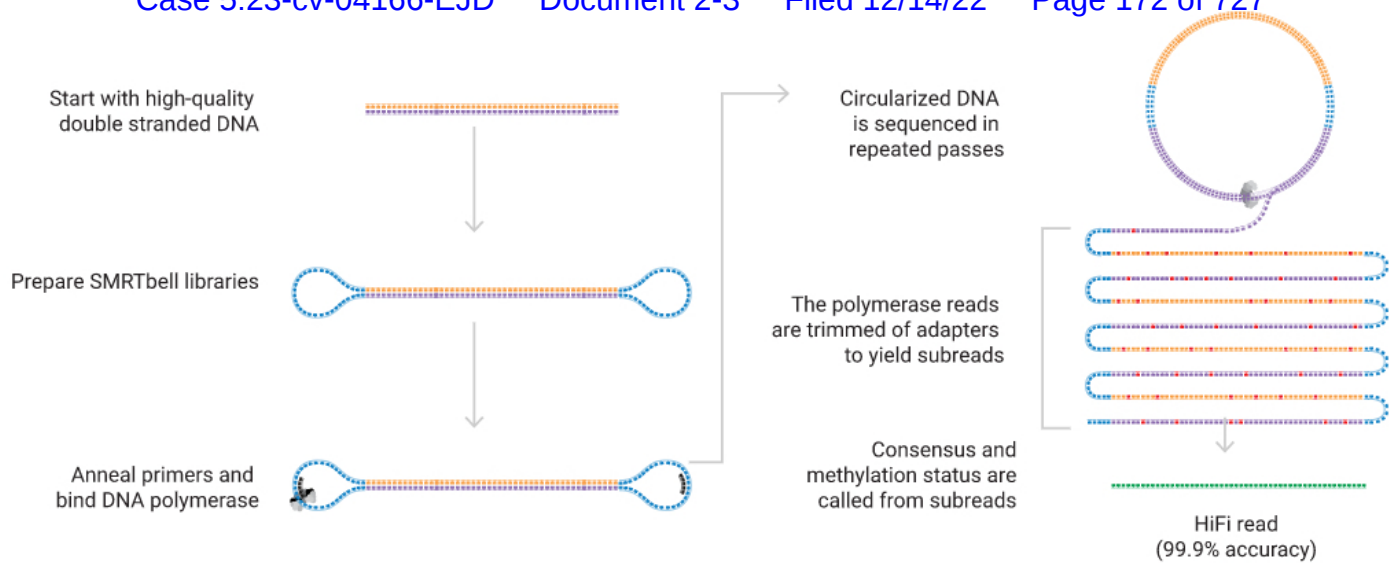


WHAT ARE HIFI READS?

HiFi reads are produced using circular consensus sequencing (CCS) mode on PacBio long-read systems. HiFi reads provide base-level resolution with 99.9% single-molecule read accuracy.

HiFi reads can be used across a wide range of SMRT sequencing applications, from whole genome sequencing for *de novo* assembly, comprehensive variant detection, epigenetic characterization, RNA sequencing and more.

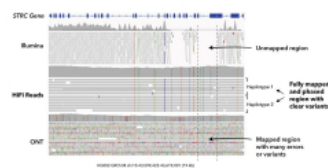
How are HiFi reads generated?



HOW DOES HIFI SEQUENCING COMPARE WITH OTHER TECHNOLOGIES?

HiFi reads let you accurately detect all types of variants, from single nucleotide to structural variants, with high precision and recall and phase haplotypes, even in hard-to-sequence regions of the genome. Explore the resources below for more information on how HiFi reads perform relative to other technologies.

Original publication: Wenger, A. M., et al. (2019) [Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome](#). *Nature Biotechnology*, 37, 1155–1162.



	SV	SV	SV
HiFi	✓	✓	✓
ILLUMINA	✓	✓	✓
ONT	✓	✓	✓

TOTAL ERRORS (SV) = (SV) / (SV)

TELOMERE-TO-TELOMERE

The *Telomere-to-Telomere* (T2T) Consortium selected HiFi sequencing to generate the first fully complete human genome assembly, 20 years after the original human genome project.

[Read more](#)

HIFI READS

HiFi reads make understanding the accuracy of your sequencing data easy and straightforward, but there is a lot to know about accuracy when considering a sequencing technology.

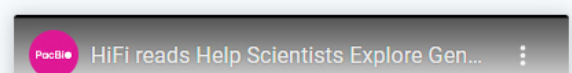
[Read more](#)

PRECISIONFDA

In the *precisionFDA Truth Challenge V2*, which evaluated methods for variant calling in human genomes, approaches that use HiFi reads delivered the highest precision and recall in all categories.

[Read more](#)

WHAT ARE THE BENEFITS OF HIFI SEQUENCING?



There are many advantages to using HiFi sequencing beyond the long read lengths and high accuracy. These include easy library preparation, low coverage requirements, small file sizes, and the fastest time to assembly. Explore the resources below to better understand these advantages.



BROCHURE

Learn more about the benefits of HiFi reads and see how they provide a single technology solution across a range of applications

[Download brochure](#)



TALK WITH AN EXPERT

If you have a question, need to check the status of an order, or are interested in purchasing an instrument, we're here to help.

First name *

Last name *

Email *

Phone

Organization *

Country *

Area of interest *

Questions *

[Connect with a PacBio scientist](#)

By registering on this web page, you are consenting and agreeing to collection and use of that information by PacBio in accordance with our [privacy policy](#).



Our revolutionary sequencing technologies combine the completeness

PRODUCTS

Technologies
Sequencing systems

FOCUS AREAS

Human genomics
Plant + animal

ENGAGE

Blog
Webinars

COMPANY

About us
Contact us

of long reads with the accuracy of short reads to provide the most comprehensive view of genomes, transcriptomes, and epigenomes.

Connect with us



Email *

Join our mailing list

Sequencing methods

Consumables

Data analysis

Purchase

Documentation

Software downloads

Infectious disease

Microbial genomics

Cancer research

Gene therapy

Events

Case studies

Sequencing 101

Access datasets

Resources

Become a service provider

Find a service provider

Find a distributor

Leadership

Press releases

Investors

Sustainability

Careers

Support

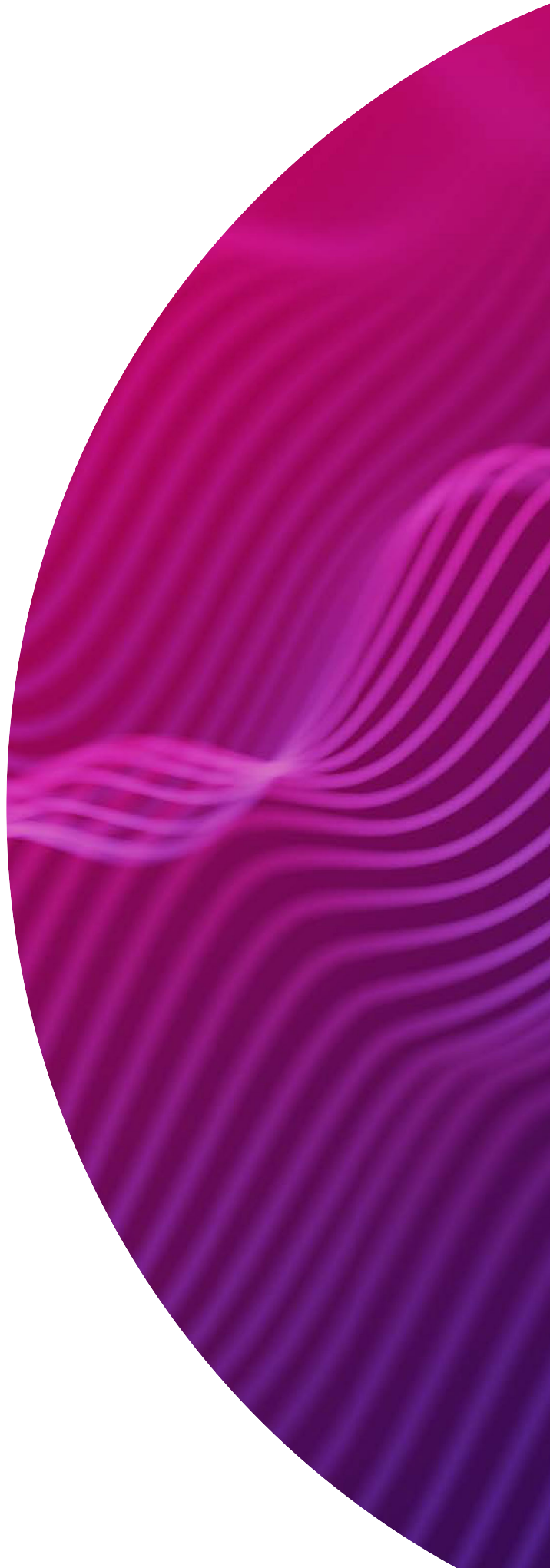
Customer login

EXHIBIT I



SMRT[®] Link user guide

Sequel[®] II and IIe
systems



Research use only. Not for use in diagnostic procedures.

P/N 102-278-200 Version 01 (April 2022)

© 2022, PacBio. All rights reserved.

Information in this document is subject to change without notice. PacBio assumes no responsibility for any errors or omissions in this document.

PACBIO DISCLAIMS ALL WARRANTIES WITH RESPECT TO THIS DOCUMENT, EXPRESS, STATUTORY, IMPLIED OR OTHERWISE, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NONINFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL PACBIO BE LIABLE, WHETHER IN CONTRACT, TORT, WARRANTY, PURSUANT TO ANY STATUTE, OR ON ANY OTHER BASIS FOR SPECIAL, CONSEQUENTIAL, INCIDENTAL, EXEMPLARY OR INDIRECT DAMAGES IN CONNECTION WITH (OR ARISING FROM) THIS DOCUMENT, WHETHER OR NOT FORESEEABLE AND WHETHER OR NOT PACBIO IS ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Certain notices, terms, conditions and/or use restrictions may pertain to your use of PacBio products and/or third party products. Refer to the applicable PacBio terms and conditions of sale and to the applicable license terms at <http://www.pacificbiosciences.com/licenses.html>.

Trademarks:

Pacific Biosciences, the PacBio logo, PacBio, Circulomics, Omnione, SMRT, SMRTbell, Iso-Seq, Sequel, Nanobind, and SBB are trademarks of Pacific Biosciences of California Inc. (PacBio). All other trademarks are the sole property of their respective owners.

See <https://github.com/broadinstitute/cromwell/blob/develop/LICENSE.txt> for Cromwell redistribution information.

PacBio
1305 O'Brien Drive
Menlo Park, CA 94025
www.pacb.com



SMRT® Link user guide (v11.0)

Introduction	4
Contact information	5
Module menu commands	6
Gear menu commands	6
Sending information to Technical Support	7
Sample Setup	8
Application-based calculations	9
Custom calculations	10
Classic mode	10
Editing or printing calculations	12
Deleting calculations	12
Importing/exporting calculations	12
Run Design	16
Creating a new Run Design	16
Custom Run Designs	18
Advanced options	19
Editing or deleting Run Designs	20
Creating a Run Design by importing a CSV file	20
Run QC	27
Table fields	29
Run settings and metrics	31
Data Management	34
What is a Data Set?	34
Creating a Data Set	35
Viewing Data Set information	36
Copying a Data Set	36
Deleting a Data Set	37
Starting a job from a Data Set	37
Data Set QC reports	37
What is a Project?	39
Data Sets and Projects	39
Creating a Project	39
Editing a Project	40
Deleting a Project	40
Viewing/deleting sequence, reference and barcode data	41
Importing sequence, reference and barcode data	41
Exporting sequence, reference and barcode data	42
SMRT® Analysis	44
Creating and starting a job	44
Starting a job after viewing sequence data	49
Canceling a running job	50

Restarting a failed job	50
Viewing job results	51
Copying and running an existing job	52
Exporting a job	52
Importing a job	52
PacBio® secondary analysis applications	54
Genome Assembly	55
HiFi Mapping	58
HiFiViral SARS-CoV-2 Analysis	62
Iso-Seq® Analysis	67
Microbial Genome Analysis	74
Minor Variants Analysis	80
Structural Variant Calling	86
PacBio® data utilities	90
5mC CpG Detection	91
Demultiplex Barcodes	92
Export Reads	98
Mark PCR Duplicates	100
Trim Ultra-Low Adapters	102
Circular Consensus Sequencing (CCS)	104
Working with barcoded data	107
Step 1: Specify the barcode setup & sample names in a Run Design	107
Step 2: Perform the sequencing run	109
Step 3: (Optional) Run the Demultiplex Barcodes data utility	110
Step 4: Run applications using the demultiplexed data as input	111
Demultiplex Barcodes details	112
Automated analysis	115
Creating an Auto Analysis job from SMRT Analysis	115
Creating Auto Analysis from a Run Design	116
HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design	116
Getting information about analyses created by Auto Analysis	116
Getting information about Pre Analysis from SMRT Analysis	117
Getting information about Pre Analysis from Run Design	117
Visualizing data using IGV	118
Using the PacBio® self-signed SSL certificate	120
Sequel® II and Sequel IIe systems output files	121
Sequel IIe system output files	121
Sequel II system output files	124
Secondary analysis output files	126
Configuration and user management	129
LDAP	129
SSL	129
Adding and deleting SMRT Link users	130

Assigning user roles 130

Hardware/software requirements..... 132

Appendix A - PacBio terminology..... 133

Appendix B - Data search 136

Appendix C - BED file format for Target Regions report 138

Appendix D - Additional information in the CCS Data Set Export report 140

Introduction

This document describes how to use PacBio's SMRT Link software. SMRT Link is the web-based end-to-end workflow manager for Sequel II systems. SMRT Link includes the following modules:

- **Sample Setup:** Calculate binding and annealing reactions for preparing DNA libraries for use on **all** Sequel II systems. (See ["Sample Setup" on page 8](#) for details.)
- **Run Design:** Design sequencing runs and create and/or import sample sheets. (See ["Run Design" on page 16](#) for details.)
- **Run QC:** Monitor run progress, status and quality metrics. (See ["Run QC" on page 27](#) for details.)
- **Data Management:** Create Projects and Data Sets; generate QC reports for Data Sets; view, import, or delete sequence, reference, and barcode files. (See ["Data Management" on page 34](#) for details.)
- **SMRT Analysis:** Perform secondary analysis on the basecalled data (such as sequence alignment, variant detection, *de novo* assembly, structural variant calling, and RNA analysis) after a run has completed. (See ["SMRT® Analysis" on page 44](#) for details.)

Note: SMRT Link v11.0 is for use with **Sequel II** systems and **Sequel IIe** systems **only**. If you are using a Sequel system, use an **earlier** version of SMRT Link.

This document also describes:

- The data files generated by the Sequel II system and Sequel IIe systems for each cell transferred to network storage. (See ["Sequel® II system and Sequel IIe system output files" on page 121](#) for details.)
- The data files generated by secondary analysis. (See ["Secondary analysis output files" on page 126](#) for details.)
- Configuration and user management. (See ["Configuration and user management" on page 129](#) for details.)
- SMRT Link client hardware/software requirements. (See ["Hardware/software requirements" on page 132](#) for details.)

Installation of SMRT Link **server** software is discussed in the document **SMRT Link software installation guide (v11.0)**.

New features, fixed issues and known issues are listed in the document **SMRT Link release notes (v11.0)**.

When you first start SMRT Link, you must specify which system you are using: **Sequel II**, or **Sequel IIe**. This choice affects some of the initial values used in the Sample Setup and Run Design modules. In those modules, you can switch between the two Sequel systems as needed. Users with administrator access can configure SMRT Link to support **all** instrument types.

Contact information

For additional technical support, contact PacBio at support@pacb.com or 1-877-920-PACB (7222).

Using SMRT® Link

You access SMRT Link using the Chrome web browser.

- SMRT Link is **not** available on the instrument – it must be accessed from a remote workstation.
- Depending on how SMRT Link was installed at your site, logging in with a user name and password may be required.
- SMRT Link needs a Secure Sockets Layer (SSL) certificate to ensure a secure connection between the SMRT Link server and your browser using the HTTPS protocol.

If an SSL certificate is **not** installed with SMRT Link, the application will use the PacBio self-signed SSL certificate and will use the HTTP protocol. In this case, **each** user will need to accept the browser security warnings described in [“Using the PacBio® self-signed SSL certificate” on page 120](#).

After accessing SMRT Link, the **home page** displays.



- Click the **PacBio logo** at the top left to navigate back to the SMRT Link home page from within the application.
- Click the **Gear menu** to sign out, configure for the Sequel II system or Sequel IIe system, view version information, or perform administrative functions (Admins **only**).
- Click a module name to access that module. **Sample Setup**, **Run Design**, **Data Management** and **SMRT Analysis** include links to create new Calculations, Run Designs, Data Sets, and jobs. (A **Module** menu displays next to the PacBio logo, allowing you to move between modules.)
- Click **?** to view the SMRT Link online help.
- Select **Sign Out** from the Gear menu to log out of SMRT Link.

Module menu commands




- **Sample Setup:** Displays the Sample Setup module.
- **Run Design:** Displays the Run Design module.
- **Run QC:** Displays the Run QC module.
- **Data Management:** Displays the Data Management module.
- **SMRT Analysis:** Displays the SMRT Analysis module.

Gear menu commands

- **Show Alarms**
 - Displays SMRT Link system-level alarms. To clear alarms, select and click **Clear Alarm** or **Clear All Alarms**.
- **Configure**
 - To specify the Sequel II system(s) that SMRT Link will be used with, click **Instruments** and check the appropriate box(es).
 - **Admin users only:** Add/delete SMRT Link users and specify their roles. See [“Adding and deleting SMRT Link users” on page 130](#) for details.
 - To specify how numbers are formatted, click **Number Formatting** and select **Period** or **Comma** as the decimal separator.
 - **(Sequel IIe system only)** To specify whether CCS analysis output includes kinetics information (used for epigenetics analysis), click **CCS Analysis Output** and select **Yes** or **No**. This is the default setting for **all** CCS analysis output, unless overwritten in individual Run Designs. **Note:** Adding kinetics information can increase the amount of storage used by the output BAM files by up to **5 times**.
- **About SMRT Link**
 - Displays software version information and available space on the server SMRT Link is connected to.
 - Click **Send** to send configuration information and/or analysis usage information to PacBio Technical Support for help in troubleshooting failed jobs.
 - **Admin users only:** 1) Update the SMRT Link **Chemistry Bundle**, which includes kit and DNA Control Complex names used in the Sample Setup and Run Design modules. 2) Update the SMRT Link **UI Bundle**, which includes changes and bug fixes to the SMRT Link Graphical User Interface or UI for a SMRT Link module.
- **Sign Out**
 - Logs you out and displays the initial login page.

Working with tables

- To sort table columns: Click a column title.
- To see additional columns: Click the > symbol next to a column title.
- To search within a table: Enter a unique search string into the Search field. (For details, see [“Appendix B - Data search” on page 136](#).)

Jobs   

Click a column name to sort on

Enter a unique search term

Displaying rows 1 to 12 out of 65

Name ▾	Member Jobs ▾	State ▾	ID ▾	Dates		Created By ▾	Job Details >
				Date Created ▾	Date Updated ▾		Workflow ↑
detect_methyl_tiny_rhino		SUCCESSFUL	176	2022-02-24, 09:28:00 AM	2022-02-24, 09:33:27 AM	smrtlinktest	SmC CpG Detection
basemods_tiny_hifi_kiwi_hpyl		SUCCESSFUL	183	2022-02-24, 09:28:11 AM	2022-02-24, 09:35:53 AM	smrtlinktest	Base Modification Analysis
basemods_tiny_ecoli_kiwi		SUCCESSFUL	196	2022-02-24, 09:28:27 AM	2022-02-24, 09:38:54 AM	smrtlinktest	Base Modification Analysis (Subre...

Sending information to Technical Support

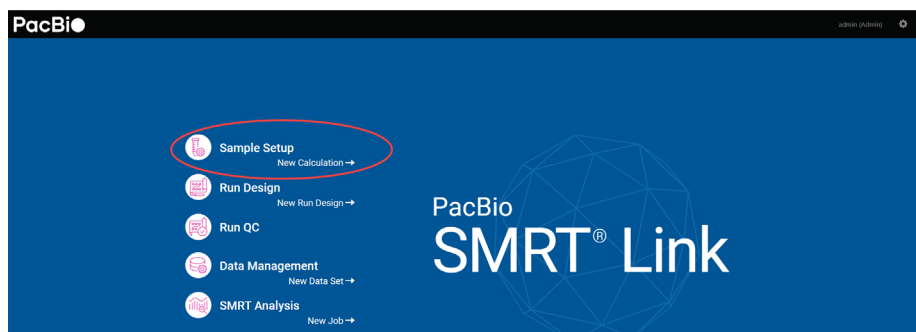
To open a case with PacBio Technical Support, send an email to support@pacb.com.

Troubleshooting information can be sent to PacBio Technical Support two ways:

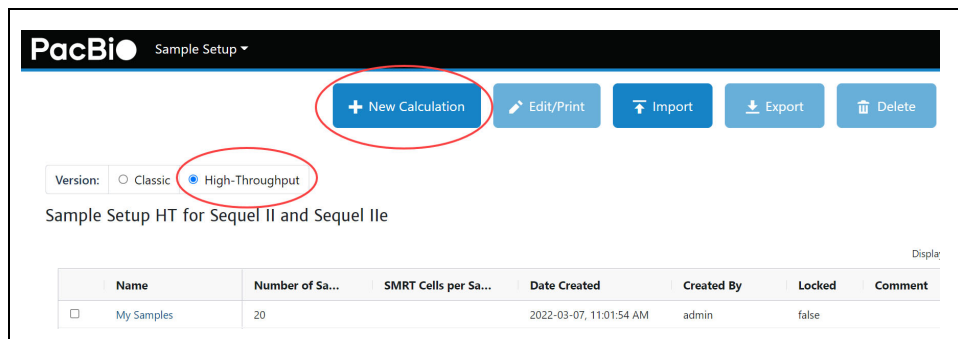
- From the SMRT Link menu: **About > Troubleshooting Information > Send.**
- From a SMRT Link “Failed” analysis Results page: Click **Send Log Files.**

Sample Setup

To prepare your samples for sequencing, use SMRT Link's **Sample Setup** module to generate a customized protocol for primer annealing and polymerase binding to SMRTbell® templates, with subsequent sample cleanup. You can then print the instructions for use in the lab.



1. Access SMRT Link using the Chrome web browser.
2. Select **Sample Setup**.
3. Select **High-Throughput** mode, which provides a more streamlined workflow to efficiently process multiple samples with similar library properties (such as mean insert size and DNA concentration) in parallel. You can also export the calculated values to a CSV file for laboratory automation.
Note: **Classic** mode is provided for legacy support purposes **only**, and is described later in this document.
4. Click **+ New Calculation**.



5. Enter the sample **name**.

Application-based calculations

6. Select a sequencing **application** for the sample. The following fields are **auto-populated** and display in green:
 - Binding Kit
 - Cleanup Anticipated Yield
7. Enter the **number of samples** for this calculation. Samples should be substantially equivalent to each other; all should have insert sizes and concentrations within +/- 15% of the specified values.
8. Enter the **number of SMRT® Cells** to bind per sample.
9. Enter the available **volume per sample**, in µL. When preparing multiple samples, this should be the **minimum** volume available for any sample.
10. Specify an **insert size**, in base pairs. The insert size is the length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters. This matches the mean insert size for the sample; the size range boundaries are described in the library preparation protocol. Enter the mean insert size of the sample(s).
11. Enter the sample **concentration(s)**, in ng/ul. Note that the acceptable range of input concentrations depends on insert size:

Insert size	Concentration range
10kb and up	20 - 60 ng/ul
3kb – 9999 bp	6 – 20 ng/ul
1.5kb – 2999 bp	3 – 10 ng/ul
500bp – 1499 bp	1 – 3 ng/ul

12. If necessary, edit the **Cleanup anticipated yield**. Adjust this percentage based on previous experience. (Cleanup removes excess primers/polymerase from bound complexes, which results in higher quality data.)
13. Specify the on-plate loading concentration (OPLC), in pM.
14. Specify the **Minimum Pipetting Volume**, in uL. This allows you to set a lower limit on pipetting volumes to use in certain protocol steps, such as sample annealing and binding. We recommend setting this to 1 uL, though in some cases, for example if sample availability is very limited, it may be appropriate to set a value below 1 uL. Some protocol steps include fixed values of 1 uL that will **not** be affected by this setting.

15. Optionally, do one of the following:
 - Click **Copy** to start a **new** sample group using the information entered. Then, edit specific fields for each sample group.
 - Click **Automate** to generate a CSV file. This exports the calculated values to a CSV file for lab automation.
16. To **print** the calculation(s) and instructions, use the browser's Print command (**Ctrl-P**).

Custom calculations

1. To accommodate new or unique sample types, choose **Application > Custom** and enter all settings manually.
2. Click **Set Custom Preset Values** to save any custom application settings you may have specified. The next time you select **Application > Custom**, those settings are retrieved.

Classic mode

Note: Classic mode is provided for legacy support purposes **only**. We **highly recommend** using High-Throughput mode even for single samples.

1. Select **Classic** mode.
2. Select **Sequel II** or **Sequel IIe**.
3. Click **+ New Calculation**.
4. Enter the sample **name**.
5. Select a sequencing **application** for the sample.
6. Enter the available sample **volume**, in μL .
7. Enter the sample **concentration**, in ng/ μL .
8. Specify an **insert size**, in base pairs.
9. Select the **Internal Control** version to use for this run from the list, or type in a part number. PacBio **highly** recommends using the Internal Control to help distinguish between sample quality and instrument issues in the event of suboptimal sequencing performance. (**Note:** PacBio **requires** the use of the Internal Control for consumables to be eligible for reimbursement consideration.)
10. If necessary, edit the **Cleanup anticipated yield**. Adjust this percentage based on previous experience. (Cleanup removes excess primers/polymerase from bound complexes, which results in higher quality data.)
11. Specify the on-plate loading concentration (OPLC), in pM.
12. Enter the number of SMRT[®] Cells to bind, at the specified on-plate loading concentration.
13. Rather than leave a small amount of library behind, use the entire library volume available if desired by selecting **Prepare Entire Sample > Yes**. This generates annealing, binding and cleanup instructions for the **entire available sample volume**. The instructions for loading the sample plate will still follow the scale indicated by the specified number of SMRT Cells to run.
14. In the complex cleanup step, enter the pre- and post-cleanup sample DNA quantitation and volume measurement results.

	Sample 1	✓
Volume of Purified Complex (uL)	<input type="text"/>	
Purified Complex Concentration (ng/uL)	<input type="text"/>	
Molar Concentration of Purified Complex (pM)	???	
Ampure Cleanup Yield (%)	???	

15. Optionally, specify an alternative number of cells or on-plate loading concentration (OPLC) for the final sample dilution step. Use this feature, for example, to initially set up a single-SMRT Cell run to test a specific loading concentration prior to conducting a multi-SMRT Cell sequencing run, or to set up a loading titration experiment to optimize the OPLC for your particular sample.

Final Loading Dilution

Reagent	Sample 1	✓	
Warning	Values measured in cleanup step must be entered		
Sequel® Complex Dilution Buffer	0.0 uL		
Prepared sample	0.0 uL		
Diluted Internal Control (Dilution 2)	0.0 uL		
DTT	0.0 uL		
Sequel Additive	0.0 uL		
Total Volume	0.0 uL		
# of SMRTCells requested	5		
Show values for a different number of cells	<input type="text"/>		
Show values for a different OPLC	<input type="text"/>		

Load 115 uL of sample per well and store at 4C for up to 24 hours before use.

16. Optionally, do one of the following:
- Click **Copy** to start a **new** sample using the information entered. Then, edit specific fields for each sample.
 - Click **Remove** to delete the current calculation.
 - Click **Lock** to lock the calculation. This is **required** before samples can be imported into the Run Design module, and also sends a finalized version of the instructions to the server for use in Data Set reports. After locking, no further changes can be made to a calculation. (Click **View** to see the locked instructions.) Locking ensures that calculations are always synchronized with their run time state if a report is generated at a later date. (**Lock** is **only** available if there are one or more samples visible **and** most fields have values entered.)
 - Click the **New Sample** button at the top of the screen to start a new, empty sample.
17. Specify whether to display the **full** instructions, or only the **loading** instructions.

Advanced options

- Specify the **Minimum Pipetting Volume**, in uL. This allows you to set a lower limit on pipetting volumes to use in certain protocol steps, such as sample annealing and binding. We recommend setting this to 1 uL, though in some cases, for example if sample availability is very limited, it may be appropriate to set a value below 1 uL. Some protocol steps include fixed values of 1 uL that will **not** be affected by this setting.
- Specify the **% of Annealing Reaction to Use in Binding**. This accommodates pipetting underage: Due to pipetting issues, volumes may not add up to what they should; a value below 100% helps ensure there will be enough annealed sample for binding.

Editing or printing calculations

1. On the **Sample Setup** screen, select one or more calculation names.
2. Click **Edit/Print**. (**Note:** If the samples use different versions of chemistry, a warning message displays.)
3. Edit the sample(s) as necessary.
4. Specify whether to display the **full** instructions, or only the **loading** instructions.
5. To print the calculation(s), use the browser's **Print** command (Ctrl-P).

Deleting calculations

1. On the **Sample Setup** screen, select one or more calculation names to delete.
2. Click **Delete**.

Importing/exporting calculations

Sample Setup supports importing and exporting calculations in CSV format.

To **import** a new calculation, first find (or create) a calculation **similar** to that you wish to import, then export it in CSV format. You can then customize the exported CSV file as needed, then **import** the modified CSV file.

Note: The content of the CSV file generated using the **Export** button in the Sample Setup home screen is **different** from the content of the CSV file generated using the High-Throughput mode's **Automate** button used for lab automation.

1. Access SMRT Link using the Chrome web browser.
2. Select **Sample Setup**.
3. Select **High-Throughput**.
4. Select an existing calculation.
5. Click **Export**, then click **Download**.
6. Edit the exported calculation in Excel (changing sample names, concentrations, and so on), then save it under a new name.

7. In Sample Setup, click **Import**.
8. Click **Browse**, then select the CSV file you previously modified in Step 6 and click **Open**. If everything is correct, click **Continue**. The imported calculation displays.

Note:

- You can select **multiple** calculations to export to the same CSV file.
- You can also **import** multiple calculations by adding rows to the CSV file.

Following are the fields contained in the CSV-format Calculations file.

Field name	Required	Description
Sample Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only .
System Name	Yes	Must be Sequel II, or Sequel IIE.
Application	Yes	Enter one of the following values: <ul style="list-style-type: none"> • HiFi Reads • Microbial Assembly • HiFiViral SARS-CoV-2 • Iso-Seq Method • Adeno-Associated Virus • Full-Length 16S rRNA Sequencing • Shotgun Metagenomic Profiling or Assembly • <3kb Amplicons • >=3kb Amplicons • Custom
Available Starting Sample Volume (uL)	Yes	Enter a positive integer. Units are in microliters.
Starting Sample Concentration (ng/uL)	Yes	Enter a positive integer. Units are in nanograms per microliter.
Insert Size (bp)	Yes	Enter a positive integer. Units are in base pairs.
Control Kit	No	Must be blank or Lxxxxx101717600123199.
Cleanup Anticipated Yield (%)	No	Enter a positive integer. Note: If Application is set to Custom , this field is required .
On Plate Loading Concentration (pM)	Yes	Enter a positive integer. Units are in parts per million.
Cells to Bind (cells)	Yes	Enter a positive integer.
Prepare Entire Sample	Yes	Enter a Boolean value: true, t, yes, y, false, f, no, or n . Boolean values are not case-sensitive.
Sequencing Primer	Yes	Enter one of the following values: <ul style="list-style-type: none"> • Sequencing Primer v2 • Sequencing Primer v4 • Sequencing Primer v5

Field name	Required	Description
Binding Kit	Yes	For Sequel II/Ile Binding Kits 2.0, 2.1, 2.2, 3.1 and 3.2: <ul style="list-style-type: none"> • Lxxxxxx101780500123199 (2.0) • Lxxxxxx101820500123199 (2.1) • Lxxxxxx101894200123199 (2.2) • Lxxxxxx102194200123199 (3.1) • Lxxxxxx102194100123199 (3.2)
Target Annealing Concentration (nM)	No	Enter a positive integer. Units are in nanomolar. Note: If Application is set to <i>Custom</i> , this field is required .
Target Binding Concentration (nM)	No	Enter a positive integer. Units are in nanomolar. Note: If Application is set to <i>Custom</i> , this field is required .
Target Polymerase Concentration (X)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Binding Time (hours)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Cleanup Bead Type	No	Must be AMPure or ProNex. Note: If Application is set to <i>Custom</i> , this field is required .
Cleanup Bead Concentration (X)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Minimum Pipetting Volume (uL)	No	Enter a positive integer. Units are in microliters.
Percent of Annealing Reaction To Use In Binding (%)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
AMPure Diluted Bound Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
AMPure Diluted Bound Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
AMPure Purified Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
AMPure Purified Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
ProNex Diluted Bound Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
ProNex Diluted Bound Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
ProNex Purified Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
ProNex Purified Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
Requested Cells Alternate (cells)	No	Enter a positive integer.
Requested OPLC Alternate (pM)	No	Enter a positive integer. Units are in parts per million.

CSV file general requirements

- Each line in the CSV file represents **one** sample.
- The CSV file may **only** contain ASCII characters. Specifically, it must satisfy the regular expression `/^[\x00-\x7F] *$/g`

Following are the fields contained in the CSV file generated by the **Automate** button in High-Throughput mode. This includes **all** the fields

that display in the Sample Setup page, with the volumes listed in each table easily accessible for liquid handling automation purposes.

Row	Field name
1	Export Version Version number of the file format specification. Allows for scripts to check version numbers to ensure compatibility through subsequent software releases.
2	Instructions Version Version number of SMRT Link, chemistry bundle, and parameters.
3	Sample Group Name
4	Annealing Number of Samples
5	Annealing Sample Volume
6	Annealing Master Mix Volume
7	Annealing Incubation Temperature (C)
8	Annealing Incubation Time (minutes)
9	Polymerase Stock Volume
10	Sequel II Polymerase Dilution Buffer Volume
11	Binding Number of Samples
12	Binding Annealed Sample Volume
13	Binding Master Mix Volume
14	Binding Diluted Polymerase Volume
15	Binding Incubation Temperature (C)
16	Binding Incubation Time (minutes)
17	ICD1 Sequel Complex Dilution Buffer Volume
18	ICD1 Internal Control Stock Volume
19	ICD2 Sequel Complex Dilution Buffer Volume
20	ICD2 Diluted Internal Control (ICD1) Volume
21	ICD3 Sequel Complex Dilution Buffer Volume
22	ICD3 Diluted Internal Control (ICD2) Volume
23	Cleanup S2 Sample Input Volume
24	Cleanup S2 Diluent Volume
25	Cleanup S2 Binding Buffer
26	Cleanup S3 Bead Solution Volume
27	Cleanup S5 Elution Volume
28	Cleanup S5 Elution Buffer
28	Final Loading Number of Samples
30	Final Loading Prepared Sample Volume
31	Final Loading Diluted Internal Control (ICD3) Volume
32	Final Loading Volume (micro-liter)

Run Design

Use SMRT Link's **Run Design** module to create, edit, or import Run Designs. A **Run Design** specifies:

- The samples, reagents, and SMRT Cells to include in the sequencing run.
- The run parameters such as movie time and loading to use for the sample.

The Run Design then becomes available from the **Sequel Instrument Control Software (ICS)**, which is the instrument touchscreen software used to select a Run Design, load the instrument, and then start the run.

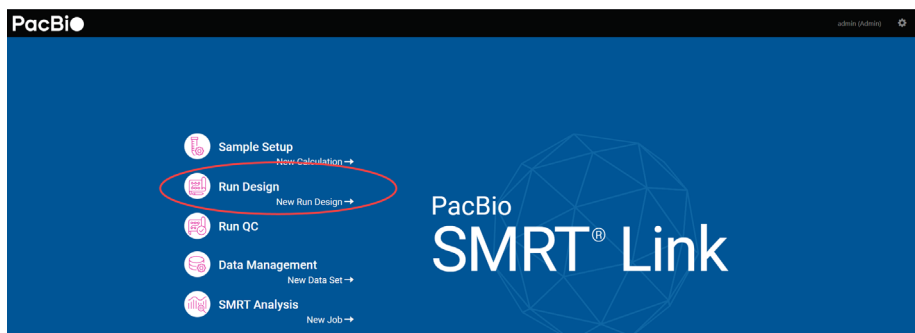
Run Designs created in SMRT Link are accessible from **all** Sequel II systems linked to the same SMRT Link server.

SMRT Link includes two different ways to create a Run Design:

- Use SMRT Link's **Run Design** module to create a new Run Design.
- Create a CSV file, then import it using SMRT Link's **Run Design** module.

Note: To create a run design, **either** use the Run Design screen, **or** import a CSV file. Do **not** mix the two methods.

Creating a new Run Design



1. Access SMRT Link using the Chrome web browser.
2. Select **Run Design**.
3. Run Designs can be sorted and searched for:
 - To sort Run Designs, click a **column title**.
 - To search for a Run Design, enter a unique search string into the **Search** field.
4. To initiate a new Run Design, click **+ Create New Design**.

5. Specify if this Run Design is to be used with a Sequel II system or a Sequel Iie system. This affects the initial default values.
6. Enter a **Run Name**. (The software creates a new run name based on the current date and time; edit the name as needed.)
7. **(Optional)** Enter **Run Comments**, **Experiment Name**, and **Experiment ID** as needed. (**Note:** Experiment ID **must** be alphanumeric.)
8. **(Optional)** Click **Select Sample** to import information from a previously-created Sample Setup entry. The following fields are auto-populated as appropriate:
 - Sample Name
 - Binding Kit
 - DNA Control Complex
 - Insert Size
 - On-Plate Loading Concentration

Application-based Run Designs

9. Select a sequencing **application** from the list. The following fields are auto-populated, and display in green:

- Template Prep Kit
 - Binding Kit
 - Sequencing Kit
 - DNA Control Complex
 - Movie Time per SMRT Cell (hours)
 - Pre-Extension Time (hours)
10. Enter a **Well Sample Name**. (This is the name of the sequencing library loaded into one well. **Example:** HG002_2019_11_02_10K)
 11. Enter a **Bio Sample Name**. (This is the name of the biological sample contained in the sequencing library, such as HG002. See [“Working with barcoded data” on page 107](#) for details.)
 12. **(Optional)** Enter **Sample Comments**.
 13. Specify the **well position** used for this sample: Click the icon to the right of the entry field and choose a plate position.
 14. Specify an **insert size** (500 base pairs minimum). The insert size is the length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters. This matches the average insert size for the sample; the size range boundaries are described in the library preparation protocol. **Note:** The default insert size for Subreads is 30,000; 10,000 for CCS reads.
 15. Specify the **On-Plate loading concentration** (OPLC), in picomolarity.
 16. **(Optional)** If you are using **barcoded samples**, see [“Step 1: Specify the barcode setup and sample names in a Run Design” on page 107](#) for instructions. For details on secondary analysis of barcoded samples, see [“Demultiplex Barcodes” on page 92](#).
 17. Sample options:
 - Click **Copy**. This starts a new sample, using the values entered in the first sample.
 - Click **Delete**. This deletes the current sample.
 - Click **Add Sample**. This starts a new, empty sample.
 18. After filling in all the samples, click **Save** - this saves the entire Run Design. The new Run Design displays on the main Run Design page.
 19. Click **View Summary** to view a table summarizing the entire Run Design. The Run Design file is now imported and available for selection in Sequel ICS on the instrument.
 20. **(Optional) Auto Analysis** allows a specific analysis to be **automatically** run after a sequencing run has finished and the data transferred to the SMRT Link server. See [“Automated analysis” on page 115](#) for details.

Custom Run Designs

To accommodate new or unique Run Designs, choose **Application > Custom** and enter all parameters manually. (See [here](#) for recommendations based on the analysis application used.)

- **Template Prep Kit, Binding Kit, or Sequencing Kit:** Select one from the list, or type in a kit part number. If the barcode is invalid, "Invalid barcode" displays.
Note: If the Sequencing or Binding kit selected is **incompatible**, an error message displays indicating the obsolete chemistry, and the run is **prevented** from proceeding.
- **DNA Control Complex:** PacBio **highly** recommends using the Internal Control to help distinguish between sample quality and instrument issues in the event of suboptimal sequencing performance. (**Note:** PacBio **requires** the use of the Internal Control for consumables to be eligible for reimbursement consideration.)
- **Movie time per SMRT Cell (hours):** Enter a time between 0.5 and 30. **Note:** The **SMRT Cell 8M** part supports **all** movie times up to 30 hours.
- **Use Pre-Extension:** If selected, optionally specify the length of pre-extension time in hours. This initiates the sequencing reaction prior to data acquisition. After the specified time, the sequencing reagents are removed from the SMRT Cell and replenished with fresh reagents, and data acquisition starts. This feature is useful for short inserts (such as ≤ 15 kb) and provides a significant increase in read length.
- **Include 5mC Calls in CpG Motifs:** If selected, analyzes the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.
- **Detect and Resolve Heteroduplex Reads:** Heteroduplexes are DNA molecules where the forward and reverse strands are not perfect reverse-complements. If the option is selected and heteroduplexes are detected, a consensus is called for **each** strand separately, and the sequence of **both** strands is output.
Note: This option displays **only** if **Adeno-Associated Virus, Full-Length 16S rRNA Sequencing, <3kb Amplicons, or ≥ 3 kb Amplicons** are selected as the application.

Advanced options

- Specify whether to use **Adaptive Loading**. Adaptive Loading uses active monitoring of the ZMW loading process to predict a favorable loading end point. Certain steps (Cleanup and Sample Dilution) require a different buffer (Adaptive Loading Buffer) if this feature is used. **Note:** Adaptive Loading **requires** the use of Sequel[®] II binding kit 2.2. If you select **Yes**, fill in the following fields:
 - **Loading Target (P1 + P2):** The fraction of ZMWs that the Adaptive Loading routine will aim to load with at least one sequencing complex. The default target for CCS applications is higher to accommodate loss of complexes during pre-extension, which is generally recommended for all CCS applications.
 - **Maximum Loading Time (hours):** This defines the maximum time the system will allow loading to progress before proceeding to sequencing. (Loading time in Adaptive Loading is flexible.)

- Specify the length of time (1, 2 or 4 hours) for **immobilization** of SMRTbell templates. This is the length of time the SMRT Cell is at the Cell Prep Station to allow diffusion of SMRTbell templates into the ZMWs. This option is **not** available if **Adaptive Loading** is selected.
 - PacBio **highly recommends** using the default immobilization time of 2 hours.
- (**Sequel IIe systems only**) Specify, for this Run Design **only**, whether to include kinetics information (used for epigenetics analysis) in the CCS analysis output. This setting **overwrites** the global setting in **Gear > Configure > CCS Analysis Output**. **Note:** Adding kinetics information can increase the amount of storage used by the output BAM files by up to **5 times**.
- Specify, for this Run Design **only**, whether to include **low quality reads** (non-HiFi reads) in the CCS analysis output. Note that this option **disables** automatic demultiplexing, 5mC detection, and heteroduplex insert detection, if applicable.
- **Add Data to Project:** Specify that Data Sets generated by SMRT Cell(s) using this Run Design be associated with the selected Project. (This also applies to any Data Sets generated using Auto Analysis. By default, **all** Data Sets are assigned to **General Project**, which is accessible to all users.)

Editing or deleting Run Designs

1. On the home page, select **Run Design**.
2. Click the name of the Run Design to edit or delete.
3. (**Optional**) Click **View Summary** to view a table summarizing the entire Run Design.
4. (**Optional**) Click **Delete** to delete the current Run Design.
5. (**Optional**) Edit any of the fields.
6. Click **Save**.

Creating a Run Design by importing a CSV file

On a remote workstation, open the sample CSV file included with the installation.

To obtain the sample CSV files

1. On the home page, select **Run Design**.
2. Click **Import Run Design**.
3. Click **Download Template**. The ZIP file containing templates (one for Sequel II systems, and one for Sequel IIe systems) downloads to your local machine.

To update and import the CSV file

1. Update the appropriate CSV file as necessary for the Run Design. (See the definitions of the Run Design attributes in the table below.)
2. Save the edited CSV file.
3. Import the file into **Sequel ICS** using SMRT Link. To do so, first access SMRT Link using the Chrome web browser.

4. Select **Run Design**.
5. Click **Import Run Design**.
6. Select the saved CSV file designed for the run and click **Open**. The file is now imported and available for selection in Sequel ICS on the instrument.

CSV file structure

- Each CSV file row represents **one sample**.
- The first row contains run-level information such as Run Name, Run Comments, and so on.
- For demultiplexed samples **only, one additional row** per barcode/Bio Sample Name combination is added below the master sample row.

Note: Specifying cluster settings configuration is **not yet** supported from the Run Design CSV

Outputting Subreads on the Sequel IIe system

The Sequel IIe system can be configured to output Subreads data in BAM format by using the Run Design CSV import mechanism. In addition to the other required columns, users can add the column **Emitted Subreads Percent** to the CSV file, with a value of 0-100 for a given collection. This results in the inclusion of Subreads from 0-100% of ZMWs in the Data Set transferred from the instrument, in a BAM file **separate** from the HiFi reads. Note that this will **not** result in the inclusion of associated scraps data for each ZMW.

Run Design attribute	Required	Description
Experiment Name	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Defaults to Run Name. Example: Standard_Edna.1
Experiment Id	No	Enter a valid experiment ID. Example: 325/3250057 <ul style="list-style-type: none"> • Experiment IDs cannot contain the following characters: <, >, :, ", \, , ?, *, or). • Experiment IDs cannot start or end with a / and cannot have two adjacent / characters, such as //. • Experiment IDs cannot contain spaces. • Specifically, Experiment IDs cannot satisfy the regular expressions: / [<>:"\\ ?*]/g, / (? : ^ \) \\ / (? : \ \$) / , / / g
Experiment Description	No	Enter any ASCII string. Defaults to Run Comments. Example: 20170530_A6_VVnC_SampleSheet
Run Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Run name must be entered for the first cell and will be applied to the remaining cells in the run. Example: 20170530_A6_VVnC_SampleSheet
System Name	No	Must be Sequel II or Sequel IIe.
Run Comments	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: ecoliK12_March2021

Run Design attribute	Required	Description
Is Collection	No	Enter a Boolean value. (See Boolean details below.) Specifies whether the row designates a Collection (TRUE) or a barcoded sample (FALSE). <ul style="list-style-type: none"> Collection lines should have the Barcode Name and Bio Sample Name fields blank. Barcoded Sample lines only need to include the Is Collection, Sample Name, Barcode Name, and Bio Sample Name fields.
Sample Well	Yes	Must be specified in every row. Well number must start with a letter A through H, and end in a number 01 through 12, i.e. A01 through H12. It must satisfy the regular expression ``/^[A-H] (? : 0 [1-9] 1 [0-2]) \$/`` Example: A01
Well Sample Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: A6_3230046_A01_SB_ChemKitv2_8rxnKit Note: The Sample Name must be unique within a run.
Movie Time per SMRT Cell (hours)	Yes	Enter a floating point number between 0.1 and 30. Time is in hours. Example: 5
Use Adaptive Loading	No	Enter a Boolean value. (See Boolean details below.)
Loading Target (P1 + P2)	No	Enter a floating point number between 0.01 and 1. Example: 0.4
Maximum Loading Time (hours)	No	Enter a floating point number between 1 and 2. Time is in hours. Example: 1.2
Sample Comment	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: A6_3230046_A01_SB_BindKit_ChemKit
Insert Size (bp)	Yes	Enter an integer ≥ 10 . Units are in base pairs. Example: 2000
On Plate Loading Concentration (pM)	No	Enter a floating point number. Units are in parts per million. Example: 5
Size Selection	No	Enter a Boolean value. (See Boolean details below.) Default is FALSE .
Template Prep Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1117100259100111716
DNA Control Complex Box Barcode	No	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1234101084300123120
Binding Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1117100862200111716
Sequencing Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM0001100861800123120
Automation Name	No	Enter <code>diffusion</code> (not case-sensitive) or a custom script. (Sequel II systems do not support magbead loading.) A path can also be used, such as <code>/path/to/my/script/my_script.py</code> . The path will not be processed further, so if the full URI is required, it must be included in the CSV, such as <code>chemistry://path/to/my/script/my_script.py</code> .
Automation Parameters	No	To enable Pre-Extension time, enter the number of hours and set the boolean value to TRUE . Example 2 hours: <code>ExtensionTime=double:2 ExtendFirst=boolean:TRUE</code> (Note: Leave blank when not using Pre-Extension time, or set the boolean value to FALSE .)

Run Design attribute	Required	Description
Detect and Resolve Heteroduplex Reads	No	Enter a boolean value. (See Boolean details below.) Set to TRUE to allow for detection of heteroduplex reads. Note: Only applicable if Application is set to one of the following: <ul style="list-style-type: none"> • Adeno-Associated Virus • <3kb Amplicons • >=3kb Amplicons • Custom
Include 5mC Calls in CpG Motifs	No	Enter a boolean value. (See Boolean details below.) Set to TRUE to allow for 5mC calls in CpG motifs. Note: Only applicable if Application is set to HiFi Reads or Custom .
Sample is Barcoded	No	Enter a boolean value. (See Boolean details below.) Set to TRUE for a barcoded run.
Demultiplex Barcodes	No	Add any of the following values: Do Not Generate, In SMRT Link, or On Instrument. If left blank, the default is Do Not Generate for all systems. Note: This is available for all applications. The following values are recommended based upon your system: <ul style="list-style-type: none"> • Sequel II system: Enter one of the following values: Do Not Generate or In SMRT Link. • Sequel IIe system: Enter one of the following values: Do Not Generate, In SMRT Link, or On Instrument.
CCS Analysis Output - Include Low Quality Reads	No	Enter a boolean value. (See Boolean details below.) <ul style="list-style-type: none"> • Set to TRUE to allow for CCS analysis with <code>--all mode activated</code> and produce a <code>reads.bam</code> file • Set to FALSE to exclude all reads with <code>rq < 0.99</code>.
Barcode Set	No	Must be a UUID for a Barcode Set present in the database. To find the UUID: Click Data Management > View Data > Barcodes . Click the Barcode file of interest, then view the UUID. Example: dad4949d-f637-0979-b5d1-9777eff62008 Note: This field is used for demultiplexed data.
Same Barcodes on Both Ends of Sequence	No	Enter a boolean value. (See Boolean details below.) Set to TRUE if symmetric, FALSE if asymmetric.
Barcode Name	No	Enter Barcode Names one per line. Example: bc1001--bc1001 <ul style="list-style-type: none"> • Use double hyphens (--) to separate the 2 barcodes of each pair. • The barcode names must be contained within the specified Barcode Set. • A given barcode name cannot appear more than once in the spreadsheet. • A maximum of 15,000 barcodes is permitted per sample.
Bio Sample Name	Yes	Enter Bio Sample Names in the same row as their associated Barcode Names. Use alphanumeric characters, spaces (allowed but not recommended for compatibility with downstream software), hyphens, underscores, colons, or periods only . Bio Sample Names cannot be longer than 40 characters. Example: sample1 Note: This field is used for collections for non-multiplexed data, and for barcoded samples in multiplexed data.

Run Design attribute	Required	Description
Pipeline ID	No	<p>Note: This field is required to create an Auto Analysis.</p> <ul style="list-style-type: none"> • 5mC CpG Detection: cromwell.workflows.pb_detect_methyl • Demultiplex Barcodes: cromwell.workflows.pb_demux_ccs • Export Reads: cromwell.workflows.pb_export_ccs • Genome Assembly: cromwell.workflows.pb_assembly_hifi • HiFi Mapping: cromwell.workflows.pb_align_ccs • HiFiViral SARS CoV-2 Analysis: cromwell.workflows.pb_sars_cov2_kit • Iso-Seq Analysis: cromwell.workflows.pb_isoseq3_ccsonly • Mark PCR Duplicates: cromwell.workflows.pb_mark_duplicates • Microbial Genome Analysis: cromwell.workflows.pb_microbial_analysis • Minor Variants Analysis: cromwell.workflows.pb_mv_ccs • Structural Variant Calling: cromwell.workflows.pb_sv_ccs • Trim Ultra-Low Adapters: cromwell.workflows.pb_trim_adapters
Analysis Name	No	<p>Enter any ASCII string. See Auto Analysis Fields below for details.</p> <p>Note: This field is required for Auto Analysis, otherwise the name will be "".</p> <p>Example: sample 1 analysis</p>
Entry Points	No	<p>Entry Points only apply to Barcode Sets and Reference Sets. In addition, this field is required for Auto Analysis.</p> <p>Enter an ASCII string in the format <code>file_type;entry_id;uuid</code>, with parameters separated by characters.</p> <ul style="list-style-type: none"> • To find the UUID: Click Data Management > View Data > HiFi Reads or Subreads. Click the Data Set of interest, then view the UUID. • See the SMRT® Tools reference guide section Appendix A - Application entry points and output files to see the entry point names for each application. <p>Example: PacBio.DataSet.BarcodeSet;eid_barcode;afe89e3f-17ca-e9b8-eae9-b701dbb1f02d PacBio.DataSet.ReferenceSet;eid_ref_dataset;6b8db144-a601-4577-ab04-ba64cad0548</p>
Task Options	No	<p>Enter an ASCII string containing the options for the application referred to in the Pipeline ID field, with parameters separated by ";" characters: <code>task_id;value_type;value</code>.</p> <p>Example: pbmm2_align.task_options.minalnlength;integer;50</p> <p>Note: This field is optional for Auto Analysis - any task options not specified will use pipeline defaults.</p>

Run Design attribute	Required	Description
Application	No	<ul style="list-style-type: none"> • HiFi Reads • Microbial Assembly • Iso-Seq Method • HiFiViral SARS-CoV-2 • Adeno-Associated Virus • Full-Length 16S rRNA Sequencing • Shotgun Metagenomic Profiling or Assembly • <3kb Amplicon Sequencing • >=3kb Amplicon Sequencing • Custom <p>If blank or contains invalid values, default is Custom.</p>
CCS Analysis Output - Include Kinetics Information	No	<p>Enter a boolean value. (See Boolean details below.) Set to <code>TRUE</code> to specify that CCS analysis output includes kinetics information (used for epigenetics analysis.) Note: Adding kinetics information can increase the amount of storage used by the output BAM files by up to 5 times.</p>

CSV file general requirements

- Each line in the CSV file represents **one** sample.
- The CSV file may **only** contain ASCII characters. Specifically, it must satisfy the regular expression `/^[\\x00-\\x7F]*$/g`

Boolean values

- Valid boolean values for **true** are: `true`, `t`, `yes`, or `y`.
- Valid boolean values for **false** are: `false`, `f`, `no`, or `n`.
- Boolean values are **not** case-sensitive.

Kit barcode requirements

Kit barcodes are composed of three parts used to make a single string:

1. Lot Number (Example: DM1234)
2. Part Number (Example: 100-619-300)
3. Expiration Date (Example: 2020-12-31)

For the above example, the full kit barcode would be:

DM1234100619300123120.

Each kit **must** have a valid Part Number and **cannot** be obsolete. The list of kits can be found through a services endpoint such as:

```
[server name]:[services port number]/smrt-link/bundles/chemistry-pb/active/
files/definitions%2FPacBioAutomationConstraints.xml
```

This services endpoint will list, for each kit, the part numbers (`PartNumber`) and whether it is obsolete (`IsObsolete`).

Dates must also be valid, meaning they must exist in the Gregorian calendar.

Auto Analysis fields

- The fields include Pipeline ID, Analysis Name, Entry Points, and Task Options.
- You can define **one** analysis for each Collection or Bio Sample. The Pipeline ID, Analysis Name and Entry Points fields are **required** to create an Auto Analysis.
- The analysis name is a concatenation of the values of the **Analysis Name** and **Bio Sample Name** fields.
- The Task Options field may be left blank; any task options not specified will use pipeline defaults.

Run QC

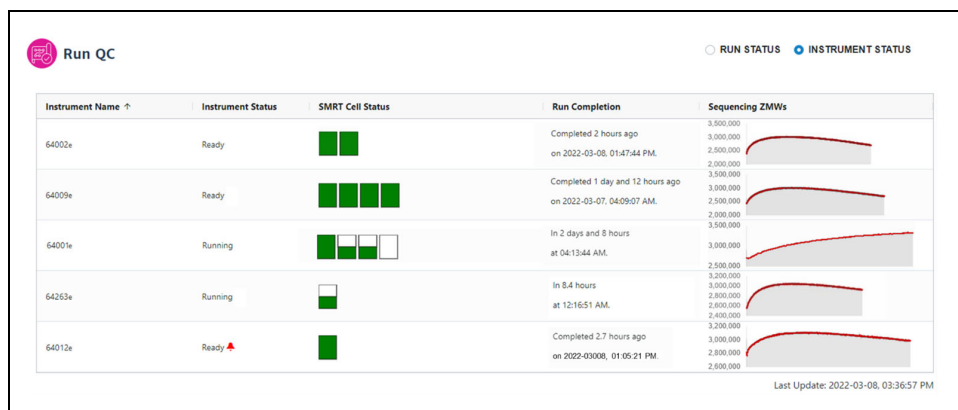
Use SMRT Link's **Run QC** module to monitor performance trends and perform run QC remotely.

Metrics can be reviewed in the Run QC module. **All** Sequel II systems connected to SMRT Link can be reviewed using Run QC.



1. Access SMRT Link using the Chrome web browser.
2. Select **Run QC**.

Accessing instrument status

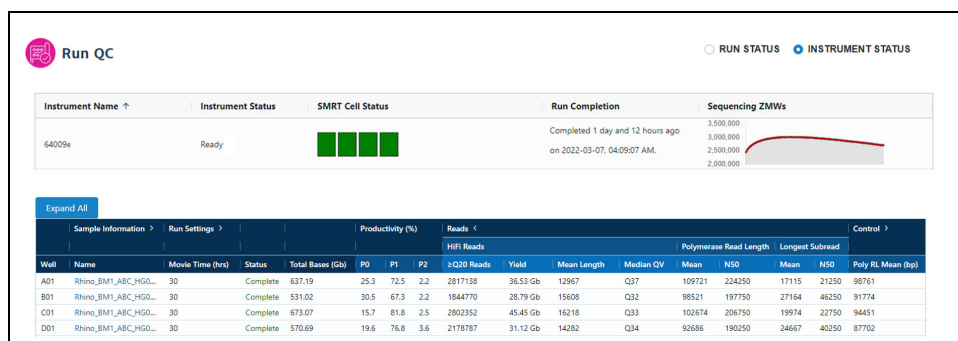


1. Select **Instrument Status**. For **each** instrument connected to the instance of SMRT Link, this displays the instrument name and its current status, SMRT Cell status, when the run will be completed, any active alarms, and how many sequencing ZMWs are active.
 - A **red alarm** symbol displays next to the instrument status if any errors or warnings appear during a sequencing run.
 - If an instrument does **not** have a SMRT Cell tray loaded, the SMRT Cell Status field will **not** display any icons. The icons are:
 - **Fully green**: The SMRT Cell has completed sequencing.
 - **Half green**: The SMRT Cell is being prepared or is currently sequencing.
 - **White**: The SMRT Cell is in the queue for sequencing, but cell preparation has not started.

- **Run Completion:** Displays the estimated time remaining to complete sequencing run or the time elapsed since the sequencing run completed. Also displays the date (in YYYY-MM-DD format) when the last sequencing run was completed.
- **Sequencing ZMWs:** Displays a plot of how many ZMWs on a SMRT Cell are actively sequencing during a movie collection. For sequencing runs conducted with Binding kit 2.2 and 3.2, **only** the number of actively sequencing singly-loaded ZMWs (P1) displays. For sequencing runs conducted with Binding kit 2.1 and 3.1, the **total** number of actively sequencing ZMWs (P1 + P2) displays.

Note: Due to terminations, not **all** ZMWs are singly-loaded at the same time. Some ZMWs are singly-loaded only **at or near the end** of a movie collection, whereas others are singly-loaded only at the **beginning**. (**Singly-loaded** means that the ZMW contains only one active polymerase instead of two or more simultaneously active polymerases.) For runs conducted with Binding kit 2.2 and 3.2, the peak concurrent **Sequencing ZMWs** value shown in the plot will always be **less** than the final %P1 ZMW yield reported in the Run QC metrics table at the end of a movie collection. (For sequencing runs conducted with Binding kit 2.1 and 3.1, the peak concurrent Sequencing ZMWs value shown in the plot will always be **higher** than the final %P1 ZMW yield reported in Run QC.) For a SMRT Cell that achieves $\geq 50\%$ P1 loading and $\geq 10\%$ P0, the ZMW Sequencing plot should typically display a peak value above 2,000,000.

See the figure below for an example comparison between the **Instrument Status report** (top) and **Run QC report** (bottom) for a WGS sample sequenced using Binding kit 3.2 with a 30-hour movie collection time. The Sequencing ZMWs plot in the Instrument Status report shows that the peak concurrent Sequencing ZMWs value for the last SMRT Cell in the run (Well D01) is approximately 3,000,000 ZMWs, whereas the final %P1 ZMW yield reported in the corresponding Run QC metrics table for Well D01 is 76.8% (or 6,144,000 P1 ZMWs.)



Accessing run information

Run QC

Sequel II Sequel IIE

Displaying rows 1 to 3 out of 3

		Dates					Instrument Details	
<input type="checkbox"/>	Name	Summary	Run Date	Completion D...	Transferred D...	Created By	Status	Instrument Na...
<input type="checkbox"/>	20210925_64011_Quokka...	20210925_64011_Quokka_JW_ML...	2021-09-25, 09:47:0...	2021-09-27, 02:22:...	2021-09-27, 05:13:...	Jellison	Complete	64011
<input type="checkbox"/>	Redwood_Bulk_5_LZ	Ecoli	2020-02-13, 07:05:5...	2020-02-17, 09:41:...	2020-02-19, 09:56:...	lthru	Complete	Sequel II Instrument
<input type="checkbox"/>	2018-07-27_64003_30kEco...	Ecoli	2018-07-27, 07:46:3...	2018-07-28, 07:30:...	2018-07-29, 06:57:...	rdalal	Complete	Sequel

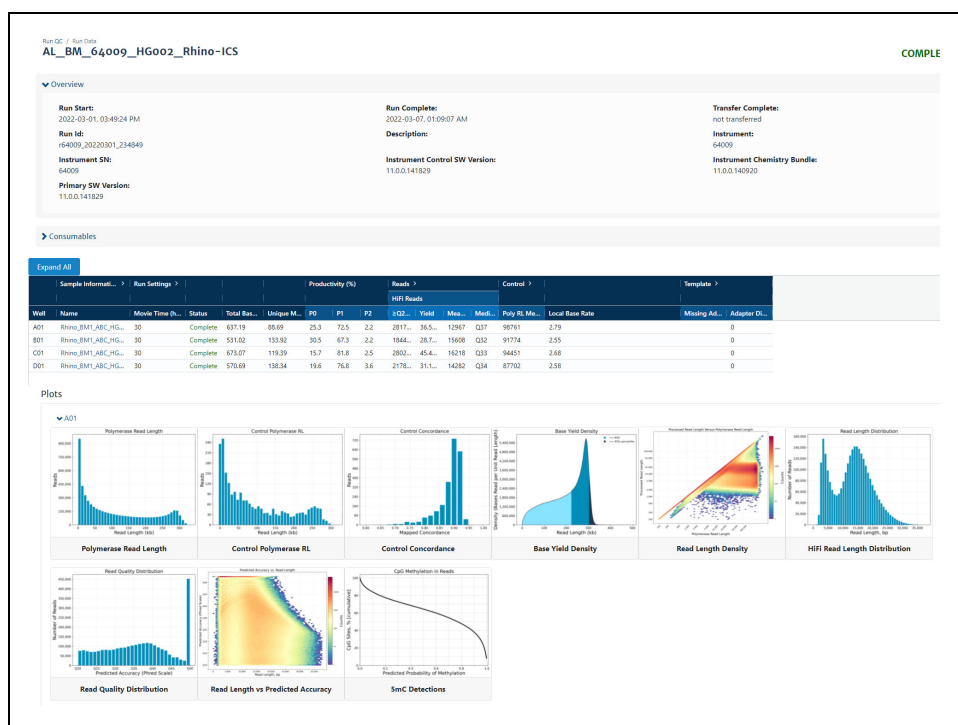
1. Select **Run Status**.
2. Click **Sequel II** or **Sequel IIE** to view informations on runs for a specific instrument model.
3. Runs can be sorted and searched for:
 - To sort runs, click a **column title**.
 - To search for a run, enter a unique search string into the **Search** field.
4. Click the run to view information about that run.
5. To **export** Run QC data in CSV format: Select one or more runs in the table, then click **Export Selected**.

Table fields

Note: Not all table fields are shown by default. To see **additional** table fields, click the > symbol next to a column title.

- **Name:** A list of all runs for the instruments connected to SMRT Link. Click a run name to view more detailed information on the individual run page.
- **Summary:** A description of the run.
- **Dates**
 - **Run Date:** The date and time when the run was started.
 - **Completion Date:** The date and time the run was completed.
 - **Transferred Date:** The date and time the run results were transferred to the network.
- **Created By:** The name of the user who created the run.
- **Status:** The current status of the run. Can be one of the following: **Running, Complete, Failed, Terminated, or Unknown**.
- **Instrument Details**
 - **Instrument Name:** The name of the instrument.
 - **Instrument SN:** The serial number of the instrument.
 - **Instrument SW:** The versions of Sequel Instrument Control Software (ICS) installed on the instrument.
- **Cells**
 - **Total:** The total number of SMRT Cells used in the run.
 - **Completed:** The number of SMRT Cells that generated data for the run.

- **Failed:** The number of SMRT Cells that failed to generate data during the run.
 - **Run ID:** An internally-generated ID number identifying the run.
 - **Primary Analysis SW:** The version of Primary Analysis software installed on the instrument.
 - **UUID:** Another internally-generated ID number identifying the run.
6. Click the **Run name** of interest. Following are the fields and metrics displayed.



- **Run Start:** The date and time when the run was started.
 - **Run Complete:** The date and time the run was completed.
 - **Transfer Complete:** The date and time that the run data was successfully transferred to the network.
 - **Run ID:** An internally-generated ID number identifying the run.
 - **Description:** The description, as defined when creating the run.
 - **Instrument:** The name of the instrument.
 - **Instrument SN:** The serial number of the instrument.
 - **Instrument Control SW Version:** The versions of Sequel Instrument Control Software (ICS) installed on the instrument.
 - **Instrument Chemistry Bundle:** The version of the Chemistry Bundle installed on the instrument when the run was initiated.
 - **Primary SW Version:** The versions of Primary Analysis software installed on the instrument.
7. Click the > arrow at the top of the **Consumables** table to see the sample wells used, consumable type, lot number, expiration date, and other information.

Run settings and metrics

Note: Click **Expand All** to expand all of the table columns. Click **Collapse All** to collapse the table columns.

- **Well:** The ID of an individual well used for this sample.
- **Sample Information**
 - **Name:** The sample name, as defined when creating the run. Clicking the name will take you to the corresponding entry in the **Data Management** module.
 - **Comment:** Sample comment, entered in Run Design.
- **Run Settings**
 - **Movie Time (hrs):** The length of the movie associated with this SMRT Cell.
 - **Loading Concentration (pM):** The on-plate loading concentration, in picomolarity.
 - **Pre-extension Time (hrs):** The pre-extension time used in the collection, if any.
 - **Workflow:** The instrument robotics workflow used for the run.
 - **Loading Time:** The time the system took for loading to progress before proceeding to sequencing.
- **Status:** The current collection status for the SMRT Cell. This can be one of the following: **Complete, Collecting, Aborted, Failed, In Progress, or Pending.**
- **Total Bases (Gb):** Calculated by multiplying the number of **productive** (P1) ZMWs by the mean polymerase read length; displayed in Gigabases.
- **Unique Molecular Yield (Gb):** The sum total length of unique single molecules that were sequenced. It is calculated as the sum of per-ZMW median subread lengths.
- **Productivity (%)**
 - **P0:** Empty ZMW; no signal detected.
 - **P1:** ZMW with a high quality read detected.
 - **P2:** Other, signal detected but no high quality read.
- **Reads:** Polymerase reads are trimmed to the high-quality region and include bases from adapters, as well as potentially multiple passes around a SMRTbell template.
 - **HiFi Reads ≥Q20 Reads:** The total number of CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Yield:** The total yield (in base pairs) of the CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Mean Length:** The mean read length of the CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Median QV:** The median number of CCS reads whose quality value is equal to or greater than 20.
 - **Polymerase Read Length Mean:** The mean high-quality read length of all polymerase reads. The value includes bases from adapters as well as multiple passes around a circular template.

- **Polymerase Read Length N50:** 50% of all read bases came from polymerase reads longer than this value.
 - **Longest Subread Mean:** The mean subread length, considering only the longest subread from each ZMW.
 - **Longest Subread N50:** 50% of all read bases came from subreads longer than this value when considering only the longest subread from each ZMW.
 - **Control**
 - **Poly RL Mean (bp):** The mean polymerase read length of the control reads.
 - **Total Reads:** The number of control reads obtained.
 - **Concordance Mean:** The average concordance (agreement) between the control raw reads and the control reference sequence.
 - **Concordance Mode:** The median concordance (agreement) between the control raw reads and the control reference sequence.
 - **Local Base Rate:** The average base incorporation rate, excluding polymerase pausing events.
 - **Template**
 - **Missing Adapter (%):** The percent of pre-filter ZMWs that are missing adapters.
 - **Adapter Dimer:** The percent of pre-filter ZMWs which have observed inserts of 0-10 bp. These are likely adapter dimers.
 - **Short Insert:** The percent of pre-filter ZMWs which have observed inserts of 11-100 bp. These are likely short fragment contamination.
8. View plots for each SMRT Cell where data was successfully transferred. Clicking on an individual plot displays an expanded view. These plots include:
- **Polymerase Read Length:** Plots the number of reads against the polymerase read length.
 - **Control Polymerase RL:** Displays the polymerase read length distribution of the control, if used.
 - **Control Concordance:** Maps control reads against the known control reference and reports the concordance.
 - **Base Yield Density:** Displays the number of bases sequenced in the collection, according to the length of the read in which they were observed. Values displayed are per unit of read length (i.e. the base yield density) and are averaged over 2000 bp windows to gently smooth the data. Regions of the graph corresponding to bases found in reads longer than the N50 and N95 values are shaded in medium and dark blue, respectively.
 - **Read Length Density:** Displays a density plot of reads, hexagonally binned according to their high-quality read length and median subread length. For very large insert libraries, most reads consist of a single subread and will fall along the diagonal. For shorter inserts, subreads

will be shorter than the HQ read length, and will appear as horizontal features. This plot is useful for quickly visualizing aspects of library quality, including insert size distributions, reads terminating at adapters, and missing adapters.

- **HiFi Read Length Distribution:** Displays a histogram distribution of HiFi reads ($QV \geq 20$), other CCS reads (three or more passes, but $QV < 20$), and other reads, by read length.
- **Read Quality Distribution:** Displays a histogram distribution of HiFi reads ($QV \geq 20$) and other CCS reads by read quality.
- **Read Length vs Predicted Accuracy:** Displays a heat map of CCS read lengths and predicted accuracies. The boundary between HiFi reads and other CCS reads is shown as a dashed line at $QV 20$.
- **5mC Detections:** If 5mC calling in CpG motifs was performed, this plot displays a reverse cumulative distribution of all detected CpG motifs according to their predicted probability of methylation.

Data Management

Use the **Data Management** module to:

- Create and manage Data Sets,
- View Data Set information,
- Create and manage Projects,
- View, import, export, or delete sequence, reference, and barcode data.

What is a Data Set?

Data Sets are logical collections of sequencing data (basecalled or analyzed) that are analyzed together, and for which reports are created. Data Sets:

- Help to **organize** and **manage** basecalled and analyzed data. This is especially valuable when dealing with large amounts of data collected from different sequencing runs from one or more instruments.
- Are the way that sequence data is represented and manipulated in SMRT Link. Sequence data from the instrument is organized in Data Sets. Data from **each** cell or collection is a Data Set.
- Can be used to collect data and summarize performance characteristics, such as data throughput, while an experiment is in progress.
- Can be used to generate reports about data, and to exchange reports with collaborators and customers.
- Can be used to start a job. (See [“Starting a job from a Data Set” on page 37](#) for details.)

A Data Set can contain sequencing data from **one** or **multiple** SMRT Cells or collections from different runs, or a portion of a collection with multiplexed samples.

For more information on Data Sets, click [here](#).

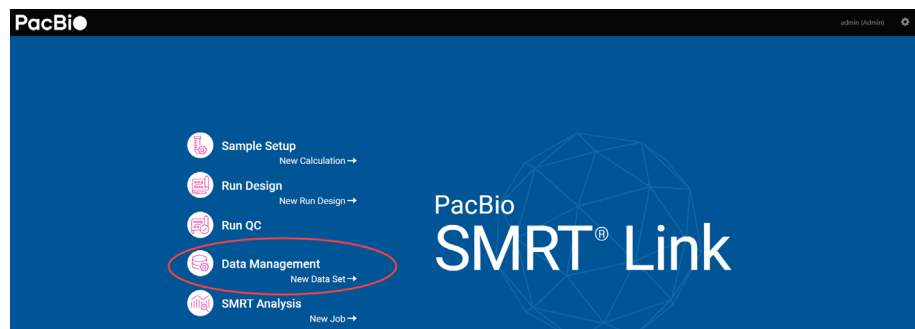
In SMRT Link, movies, cells/collections, context names and well samples are all in one-to-one relationships and can be used more or less interchangeably. That is, a Data Set from a single cell or collection will also be from a single collection derived from DNA from a single well sample. Data produced by SMRT Cells, however, can be used by **multiple** Data Sets, so that data may have a many-to-one relationship with collections.

Some Data Sets can contain **basecalled** data, while others can contain **analyzed** data:

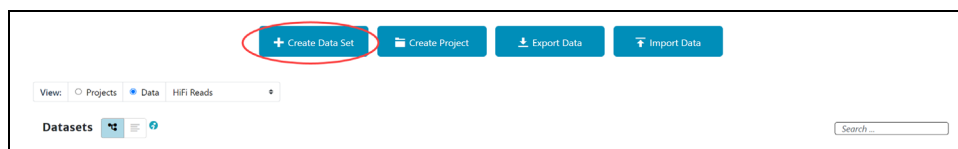
- **Basecalled data** Data Sets contain sequence data from one or multiple cells or collections.
- **Analyzed data** Data Sets contain data from previous analyse(s).

Elements within a Data Set are of the same data type, typically subreads or consensus reads, in aligned or unaligned format.

Creating a Data Set



1. Access SMRT Link using the Chrome web browser.
2. Select **Data Management**.
3. Data Sets can be sorted and searched for:
 - To sort Data Sets, click a **column title**.
 - To search for a Data Set, use the Search function. See [“Appendix B - Data search” on page 136](#) for details.



4. Click **+ Create Data Set**.
5. Enter a name for the new Data Set.

Create Data Set

Cancel Save Data Set

Data Set Name Required: Data_Set_98

Well Sample Name Required: SMS_Kiwi_PA_S4006_3280182_A01_lambdaBsaA1_Diffusion_0.75pM

Bio Sample Name Required: SMS_Kiwi_PA_S4006_3280182_A01_lambdaBsaA1_Diffusion_0.75pM

Filter Reads by QV ≥ ☐
 Filter Reads by Length bp to bp ☐

Data Type: HiFi Reads

Datasets

Data Set Details				Sample Details			
Name	Well Sample Name	Run Name	Date Created	Created By	Bio Sample Name	Barcode Name	Total Length of Read
Tiny Lambda m54006...	SMS_Kiwi_PA_S4006_3...	20180706_A6_Kiwi...	2021-10-06, 11:28...	smrtlinktest	SMS_Kiwi_PA_S4006_3...		284,298

6. Select the type of data to include in the new Data Set:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
7. (Optional) Specify the **Project** that this new Data Set will be associated with using the **Projects** menu (located at the top-right of

the Data Management page.) **General Project:** This Data Set will be visible to **all** SMRT Link users. **All My Projects:** This Data Set will be visible **only** to users who have access to Projects that you are a member of.

Note: Selecting a Project **also** filters the Data Sets that you can use when **creating** the new Data Set.

8. In the **Data Sets** table, select one or more sets of sequence data.
9. **(Optional)** Choose how to **view** the Data Set table: 1) Tree Mode - A barcoded Data Set displays as **one row**. 2) Flat Mode - A barcoded Data Set and its demultiplexed subsets display as **separate rows**.
10. **(Optional)** Use the Search function to search for specific Data Sets. See ["Appendix B - Data search" on page 136](#) for details.
11. **(Optional)** If you selected **one** Data Set **only**, click the **Filter Reads by Length** box above the Data Set list. Enter the minimum and/or maximum length to retain in the new Data Set.
12. **(Optional)** If you selected **one** Data Set **only**, click the **Filter Reads by QV₂** box above the Data Set list. Enter the minimum quality value to retain in the new Data Set.
13. Click **Save Data Set**. The new Data Set becomes available for starting analyses, viewing, or generating reports.
14. After the Data Set is created, click its name in the main Data Management screen to see reports, metrics, and charts describing the data included in the Data Set. See ["Data Set QC reports" on page 37](#) for details.



Viewing Data Set information

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of Data Set to view:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See ["Appendix B - Data search" on page 136](#) for details.
4. Click the name of the Data Set to see information about the sequence data included in the Data Set, as well as QC reports.

Copying a Data Set

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of data to copy:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.

- **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.

3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 136](#) for details.
4. Click the name of the Data Set to copy. The Data Set Reports page displays.
5. Click **Copy**. The main Data Management page displays; the new Data Set has **(copy)** appended to the name.

Deleting a Data Set

Note: SMRT Link's Delete Data Set functionality deletes the Data Set from the SMRT Link interface **only**, **not** from your server.

It is good practice to export Data Sets you no longer need to a backup server, then delete them from SMRT Link. This frees up space in the SMRT Link interface.

1. On the home page, select **Data Management**.
 2. Click **View > Data** and select the type of data to delete:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
- The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 136](#) for details.
 4. Click the name of the Data Set to delete.
 5. Click **Delete**. Note that this deletes the Data Set from the SMRT Link interface **only**; **not** from your server. To delete the Data Set from your server, **manually** delete it from the disk.
 6. Click **Yes**. The Data Set is no longer available from SMRT Link.

Starting a job from a Data Set

From the Data Set reports page, a job can be started using the Data Set.

1. Click **Analyze...**, then name the job and click **Next**.
2. Follow the instructions starting at Step 12 of [“Creating and starting a job” on page 44](#).

Data Set QC reports

The Data Set QC reports are generated when you create a new Data Set or update the data contained in existing Data Sets. These reports are designed to provide all relevant information about the data included in the Data Set as it comes from the instrument prior to data analysis, and are useful for data QC purposes.

The following reports are generated by default:

Dataset Overview

Status

Dataset Tiny Kiwi ecoli 3-plex (CCS)

Dataset ID 28

Dataset UUID a3180f93-496b-43ed-1c21-3aacc34d3b75

Well Sample Name SMS_Kiwi_Verif_54043_3280212_A01_VVCT158_28_384plex_jobx40_Diffusion_2.5pM

Biological Sample Name [multiple]

Description CCS dataset converted

Number of Records 20

Total Length 47,113

Status SUCCESSFUL

Date Created 2018-10-01, 10:51:33 AM

Date Imported 2020-09-24, 10:13:54 AM

Date Updated 2020-09-24, 10:15:00 AM

Job ID 39

Data Path /jpb/dept/secondary/bv/testdata/SAS-Sequel/ecoli/54043_20180917_210804/1_A01_micro/m54043_180917_211649_micro.consensusreadset.xml

Run name 20180917_43_Kiwi_Verif

Cell Index 0

Cell Id BA343109

Instrument Name Flex43

Well Name A01

Metadata Context Id m54043_180917_211649

Data Set Overview > Status

Displays the following information about the Data Set:

- The Data Set Name, ID, description, and when it was created and updated.
- The number of reads and their total length in base pairs.
- The names of the run and instrument that generated the data.
- The biological sample name and well sample names of the sample used to generate the data.
- Path to the location on your cluster where the data is stored, which can be used for command-line navigation. For information on command-line usage, see **SMRT® Tools reference guide (v11.0)**.

Completed Analyses

Lists all completed analyses that used the Data Set as input. To view details about a specific analysis, click its name.

Raw Data Report > Summary Metrics

- **Polymerase Read Bases:** The total number of polymerase read bases in the Data Set.
- **Polymerase Reads:** The total number of polymerase reads in the Data Set.
- **Polymerase Read Length (mean):** The mean read length of all polymerase reads in the Data Set.
- **Polymerase Read N50:** The read length at which 50% of all the bases in the Data Set are in polymerase reads longer than, or equal to, this value.
- **Subread Length (mean):** The mean read length of all subreads in the Data Set.
- **Subread N50:** The length at which 50% of all the subreads in the Data Set are longer than, or equal to, this value.
- **Insert Length (mean):** The mean length of all the inserts in the Data Set.
- **Insert N50:** The length at which 50% of all the inserts in the Data Set are longer than, or equal to, this value.

Information on loading, control reads, and adapters is also displayed. Other information may display based on the Data Set type.

What is a Project?

- Projects are collections of Data Sets, and can be used to restrict access to Data Sets to a subset of SMRT Link users.
- By default, **all** Data Sets and data belong to the **General Project** and are accessible to **all** users of SMRT Link.
- **Any** SMRT Link user can create a Project and be the owner. Projects must have an owner, and can have **multiple** owners.
- Unless a Project is shared with other SMRT Link users, it is **only** accessible by the owner.
- Only owner(s) can delete a Project; deleting a Project deletes **all** Data Sets and analyses that are part of the Project.

Projects include:

- One or more Data Sets and associated Quality Control information.
- One or more analysis results and the associated Data Sets, including information for all analysis parameters and reference sequence (if used).

Data Sets and Projects

- Once created, a Data Set **always** belongs to at least **one** project; either the **General** project or another project the user has access to.
- Data Sets can be associated with **multiple** projects.
- The data represented by a Data Set can be copied into **multiple** projects using the Data Management report page **Copy** button. Any changes made to a particular copy of a Data Set affect **only** that copy, **not** any other copies in other Projects. If a Data Set is to be used with multiple Projects, PacBio recommends that you make a **separate copy** for each Project.
- Use the **Projects** menu (located at the top-right of the Data Management page) to filter the Data Sets displayed; this is based on which Projects the Data Sets are associated with.

Creating a Project

The screenshot shows the 'Create Project' form in the SMRT Link interface. The form is titled 'Data Management / Projects' and 'Create Project'. It includes a 'Project Name' field with the value 'QC_Group', a 'Description' field, and an 'Associated Data Sets' section with a 'Select Data Sets' button. The 'Members' section has dropdowns for 'Access for All SMRT® Link users' (set to 'None') and 'Access for Individual SMRT® Link Users'. Below these are two rows of users: 'Administrator (Administrator@p...' and 'EPMAAdmin2', each with a 'View' button. A search bar 'QA' is present, and a table lists users with checkboxes. The 'QA' user is selected. A 'Search By Name' button is also visible. At the bottom right, there are 'Cancel' and 'Save' buttons.

1. Access SMRT Link using the Chrome web browser.

2. Select **Data Management**.
3. Click **+ Create Project**.
4. Enter a name for the new project.
5. **(Optional)** Enter a description for the project.
6. Click **Select Data Sets** and select one or more sets of sequence data to associate with the project.
 - **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 136](#) for details.
7. **(Optional)** Share the Project with other SMRT Link users. **(Note:** Unless a Project is shared, it is **only** visible to the owner.) There are two ways to specify who can access the new Project, using the controls in the **Members** section:
 - **Access for all SMRT Link Users: None** - No one can access the project other than the user who created it; **View** - Everyone can view the Project; **View/Edit**: Everyone can see and edit the Project.
 - **Access for Individual SMRT Link Users:** Enter a user name and click **Search By Name**. Choose **Owner**, **View**, or **View/Edit**, then click **Add Selected User**.
 - **Notes:** A) Projects can have **multiple** owners. B) If you enable **all** SMRT Link users to have **View/Edit** access, you cannot change an individual member's access to **View**.
8. Click **Save**. The new project becomes available for SMRT Link users who now have access.

Editing a Project

1. On the home page, select **Data Management**.
2. Click **View > Projects**.
3. Projects can be sorted and searched for:
 - To sort Projects: Click a **column title**.
 - To search for a Project, use the Search function. See [“Appendix B - Data search” on page 136](#) for details.
4. Click the name of the project to edit.
 - **(Optional)** Edit the Project name or description.
 - **(Optional)** Delete a Data Set associated with the Project: Click **X**.
 - **(Optional)** Add one or more sets of sequence data to the Project: Click **Select Data Sets** and select one or more Data Sets to add.
 - **(Optional)** Delete members: Click **X** next to a Project member's name to delete that user from access to the Project.
 - **(Optional)** Add members to the Project: See Step 7 in **Creating a Project**.
5. Click **Save**. The modified Project is saved.

Deleting a Project

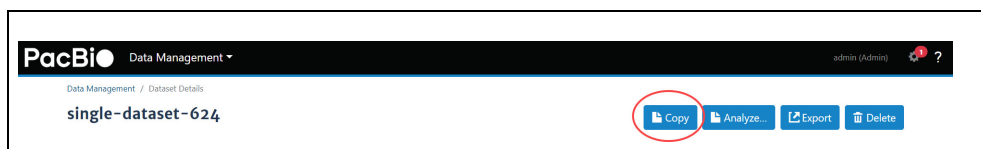
1. On the home page, select **Data Management**.
2. Click **View > Projects**.
3. Click the name of the Project to delete.

4. Click **Delete**. (This deletes **all** Data Sets and analyses that are part of the Project from SMRT Link, but **not** from the server.)

Viewing/deleting sequence, reference and barcode data

1. On the home page, select **Data Management**.
2. Click **View > Data**, then choose the type of data to view or delete:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
 - **Barcodes**: Barcodes from barcoded samples.
 - **References**: Reference sequence FASTA files used when creating certain analyses.
3. (Optional) Use the Search function to search for specific Data Sets, barcode files or reference sequence files. See [“Appendix B - Data search” on page 136](#) for details.
4. Click the name of the sequence, reference or barcode file of interest. Details for that sequence, reference sequence file or barcode file display.
5. (Optional) To delete the sequence data, reference sequence, or barcode file, click **Delete**.

Note: The **Copy** button is available for Subreads and HiFi reads, but **not** for Reference and Barcode data.



Importing sequence, reference and barcode data

Note: If your Sequel II system or Sequel IIe system is linked to the SMRT Link software during the instrument installation, your instrument data will be **automatically** imported into SMRT Link.

Several types of sequence data, as well as barcode files, can be imported for use in SMRT Link.

1. On the home page, select **Data Management**.
2. Click **Import Data**.
3. Specify whether to import data from the **SMRT Link Server**, or from a **Local File System**. (**Note: Only** references and barcodes are available if you select **Local File System**.)

Data Management / Import

Import Data

Import from

SMRT Link Server

Local File System

✕ Cancel

Select File

Data Type

- HiFi Reads or Subreads (XML)
- HiFi Reads or Subreads (ZIP)
- Barcodes (FASTA)
- Barcodes (XML)
- Barcodes (ZIP)
- References (FASTA)
- References (XML)
- References (ZIP)

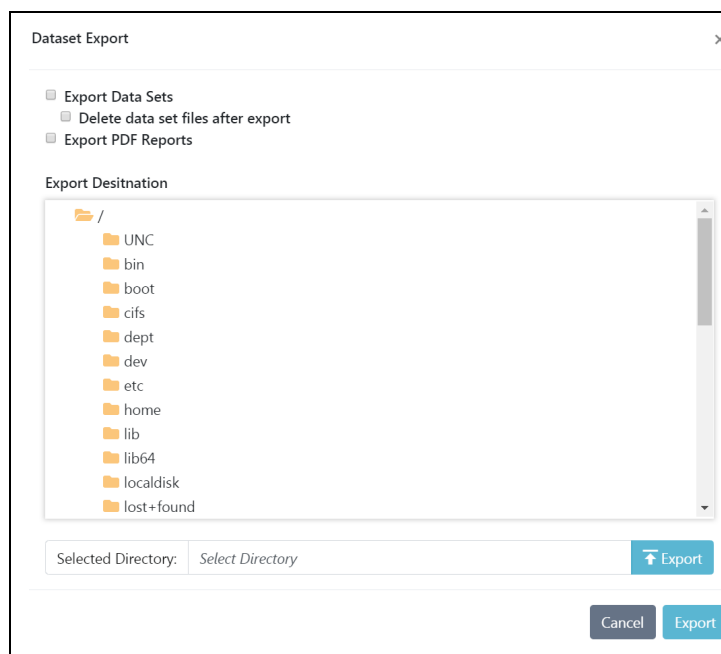
4. Select the data type to import:
 - **Subreads:** XML file (`.subreadset.xml`) or ZIP file containing information about subreads from Sequel II systems, such as paths to the BAM files.
Use **only** ZIP files created by SMRT Link.
 - **HiFi reads:** XML file (`.consensusreadset.xml`) or ZIP file containing information about HiFi reads (reads generated with CCS analysis whose quality value is equal to or greater than 20.)
Use **only** ZIP files created by SMRT Link.
 - **Barcodes:** FASTA (`.fa` or `.fasta`), XML (`.barcodeset.xml`), or ZIP files containing barcodes.
 - **References:** FASTA (`.fa` or `.fasta`), XML (`.referenceSet.xml`), or ZIP files containing a reference sequence for use in starting analyses. (**Note:** If importing from a **local system**, Reference files must be smaller than 15 MB.)
 - **Note:** FASTA files imported into SMRT Link must **not** contain empty lines or non-alphanumeric characters. The file name must **not** start with a number. For information about the file types listed here, click [here](#).
5. Navigate to the appropriate file and click **Import**. The sequence data, reference, or barcodes are imported and becomes available in SMRT Link.

Exporting sequence, reference and barcode data

Two types of sequence data (HiFi reads and Subreads) can be exported, as well as barcode files and reference files.

1. On the home page, select **Data Management**.
2. Click **Export Data**.
3. Select the type of data to export:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

- **Barcodes:** Files containing barcodes.
 - **References:** Files containing a reference sequence for use in starting analyses.
4. **(Optional)** Use the Search function to search for Data Sets, barcode files, or reference files. See [“Appendix B - Data search” on page 136](#) for details.
 5. Select one or more sets of data to export. (Multiple data files are combined as one ZIP file for export.)
 6. Click **Export Selected**.

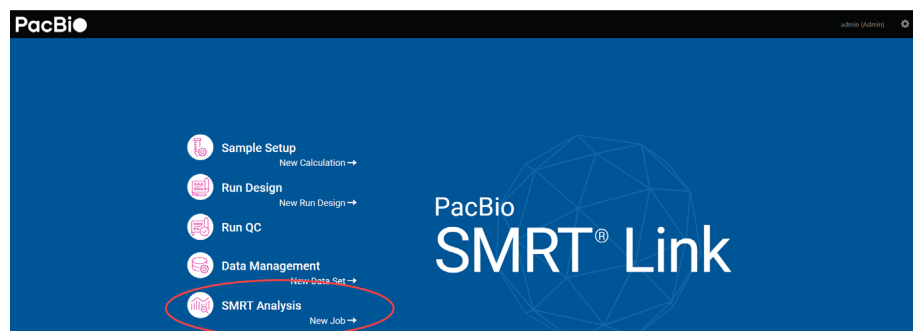


7. Navigate to the export destination directory.
8. **(Optional)** If exporting Data Sets, click **Delete data set files after export** to delete the Data Set(s) you selected from the SMRT Link installation. (Exporting, then deleting, Data Sets is useful for archiving Data Sets you no longer need.)
9. **(Optional)** If exporting Data Sets, click **Export PDF Reports** to create PDF files containing comprehensive information about the Data Set(s). Each PDF report contains extensive information about one Data Set, including loading statistics, run set up and QC information, analysis parameters and results including charts and histograms, and lists of the output files generated, all in one convenient document.
10. Click **Export**.

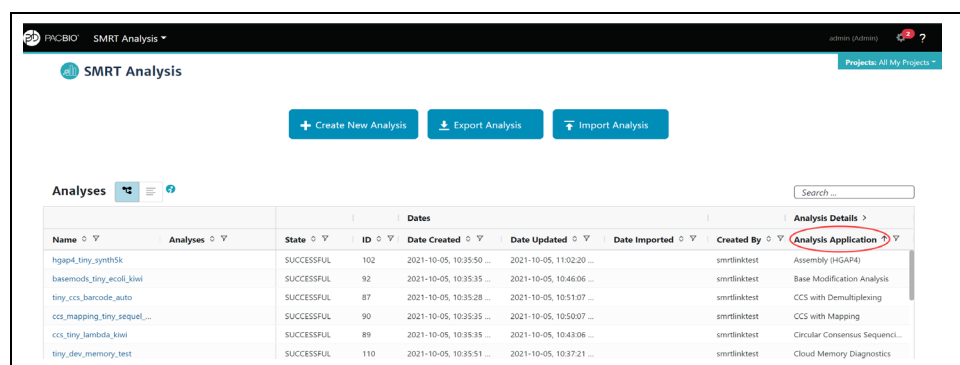
SMRT® Analysis

After a run has completed, use SMRT Link's **SMRT Analysis** module to perform **secondary analysis** of the data.

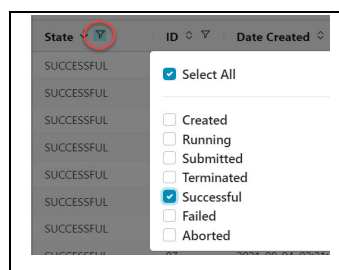
Creating and starting a job



1. Access SMRT Link using the Chrome web browser.
2. Select **SMRT Analysis**.



3. Jobs can be sorted, searched for, and filtered:
 - To **sort** jobs, click a **column title**.
 - To **search** for a job, use the Search function. See [“Appendix B - Data search” on page 136](#) for details.)
 - To **filter** the list of jobs based on their **state**: Click the funnel in the **State** column header, then click one or more of the categories of interest: **Select All**, **Created**, **Running**, **Submitted**, **Terminated**, **Successful**, **Failed**, or **Aborted**.

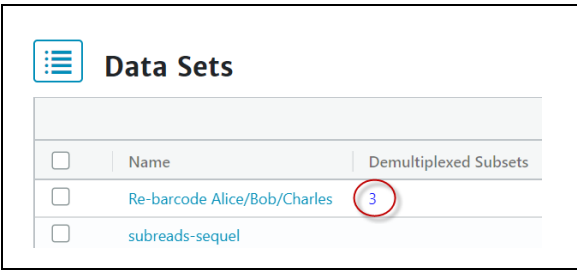


- To filter the list of jobs based on the **Project(s)** that they are associated with: Click the **Projects** menu (located at the top-right of the main SMRT Analysis page) and select a Project. See [“What is a Project?” on page 39](#) for details.
- 4. Click **+ Create New Job**.
- 5. **(Optional)** Click **Copy From...**, choose a job whose settings you wish to reuse, then click **Select**. The job name and the Data Type are filled in. Go to Step 10 to select Data Set(s).
- 6. Enter a **name** for the job.
- 7. Specify the type of job to create:
 - **Analysis** - Uses applications designed to produce biologically-meaningful results. These applications **only** accept HiFi reads.
 - **Auto Analysis** - For information on the Auto Analysis feature, see [“Automated analysis” on page 115](#) for details.
 - **Data Utility** - Data processing utilities used as **intermediate steps** to producing biologically-meaningful results.
- 8. If you selected **Data Utility**, select the type of data to use for the job:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
- 9. **(Optional)** Specify the **Project** that this job will be associated with using the **Projects** menu (located at the top-right of the SMRT Analysis page.) **General Project**: This job will be visible to **all** SMRT Link users. **All My Projects**: This job will be visible **only** to users who have access to Projects that you are a member of. To **restrict access** to a job, make sure to select a Project limited to the appropriate users **before** starting the job.

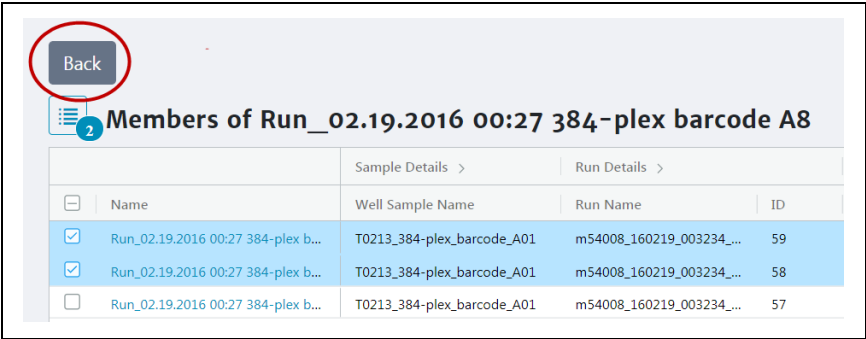
Note: Selecting a Project **also** filters the Data Sets that you can use when **creating** the job.

- 10. In the **Data Sets** table, select one or more sets of data to be analyzed.
 - **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 136](#) for details.)
 - **(Optional)** Choose how to **view** the Data Set table: 1) Tree Mode - A barcoded Data Set displays as **one row**. 2) Flat Mode - A barcoded Data Set and its demultiplexed subsets display as **separate rows**.
 - **(Optional)** For Data Sets that include demultiplexed subsets, you can also select individual subsets as part of your selection. To do so:

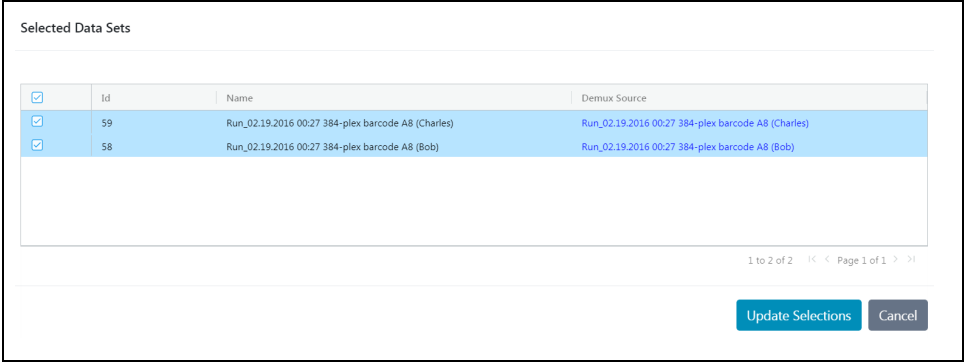
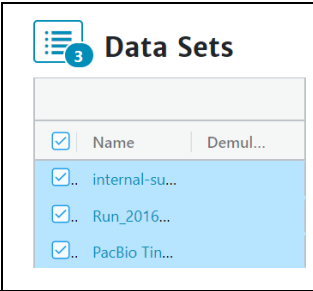
A) Click the Demultiplexed Subsets number link:



B) Select one or more subsets, then click **Back**:



C) Click the list image to view or edit the full Data Set selection. (The small blue number specifies how many Data Sets and/or subsets were selected):



Note: For information on the Auto Analysis feature, see “Automated analysis” on page 115 for details.

11. If you selected **multiple** Data Sets as input for the job, additional options become available:

Workflow Type

☒ ANALYSIS ☐ AUTO ANALYSIS ☐ DATA UTILITY

Analysis of Multiple Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis for All Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis per Data Set - Custom Parameters

- **One Analysis for All Data Sets:** Runs one job using all the selected Data Sets as input, for a maximum of 30 Data Sets.
 - **One Analysis per Data Set - Identical Parameters:** Runs one separate job for **each** of the selected Data Sets, using the **same** parameters, for a maximum of 10,000 Data Sets. Later in the process, optionally click **Advanced Parameters** and modify parameters.
 - **One Analysis per Data Set - Custom Parameters:** Runs one separate job for **each** of the selected Data Sets, using **different** parameters for each Data Set, for a maximum of 16 Data Sets. Later in the process, click **Advanced Parameters** and modify parameters. Then click **Start and Create Next**. You can then specify parameters for **each** of the included Data Sets.
 - **Note:** The number of Data Sets listed is based on testing using PacBio's suggested compute configuration, listed in **SMRT Link software installation guide (v11.0)**.
12. Click **Next**.
13. Select a secondary analysis application or data utility from the drop-down list. (Different choices display based on your initial choice of **Analysis** or **Data Utility** in Step 7. See "[PacBio® secondary analysis applications](#)" on page 54 or "[PacBio® data utilities](#)" on page 90 for details.)

Analysis Application Required

Genome Assembly

HiFi Mapping

HiFiViral SARS-CoV-2 Analysis

Iso-Seq Analysis

Microbial Genome Analysis

Minor Variants Analysis

Structural Variant Calling

- Each of the secondary analysis applications/data utilities has **required parameters** that are displayed. Review the default values shown.

- Secondary analysis applications/data utilities also have **advanced parameters**. These are set to default values, and need only be changed when analyzing data generated in non-standard experimental conditions.

14. (Optional) Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application or data utility. The imported settings are set.

The **Iso-Seq** application will be used as an example. This application characterizes full-length transcript isoforms.

15. Click the **Reference Set** field and select a reference sequence from the dialog. (The reference sequences available in SMRT Link and displayed in the dialog were imported into SMRT Analysis. See [“Importing sequence, reference and barcode data” on page 41](#) for details.)

SMRT Analysis / Create New Analysis

1. Select Data 2. Select Analysis

Analysis Application Required
Iso-Seq Analysis

Analysis Name
bbb

Import Analysis Settings Export

Associated Inputs

Primer Set Required
IsoSeqPrimers_v2

Reference Set

Run Clustering
ON OFF

Cluster Barcoded Samples Separately
ON OFF

Advanced Parameters

Analysis Datasets

I..	Name
111	single-dataset-449

16. (Optional) Click **Advanced Parameters** and specify the values of the parameters you would like to change. Click **OK** when finished. (Different applications/data utilities have different advanced parameters.)
 - To see information about parameters for **all** secondary analysis applications and data utilities provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#) and [“PacBio® data utilities” on page 90](#).

Advanced Analysis Parameters

Min. CCS Predicted Accuracy (Phred Scale) ⓘ <input type="text" value="20"/>	Require and Trim Poly(A) Tail ⓘ <input checked="" type="radio"/> ON <input type="radio"/> OFF	Minimum Mapped Length (bp) ⓘ <input type="text" value="50"/>
Minimum Gap-Compressed Identity (%) ⓘ <input type="text" value="95"/>	Minimum Mapped Coverage (%) ⓘ <input type="text" value="99"/>	Maximum Fuzzy Junction Difference (bp) ⓘ <input type="text" value="5"/>
Filters to Add to the Data Set ⓘ <input type="text"/>	Advanced pbmm2 Options ⓘ <input type="text"/>	Compute Settings ⓘ -- select --

17. **(Optional)** Click **Export** to create a CSV file containing **all** the settings you specified for the application/data utility. You can then import this file when creating future jobs using the same application/data utility. You can also use this exported file as a template for use with later jobs.
18. **(Optional)** Click **Back** if you need to change any of the analysis attributes selected in Step 7.
19. Click **Start** to submit the job. (If you selected multiple Data Sets as input, click **Start Multiple Jobs** or **Start and Create Next**.)
20. Select **SMRT Analysis** from the Module Menu to navigate to the main SMRT Analysis screen. There, the status of the job displays. When the job has **completed**, click on its name - reports are available for the completed job.
21. **(Optional)** To **delete** the completed job: Click **Delete**, then click **Yes** in the confirmation dialog. The job is deleted from **both** the SMRT Link interface and from the server.

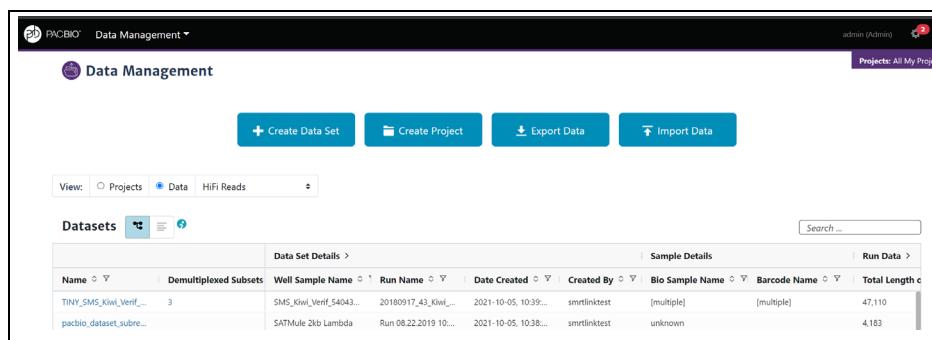
The screenshot shows the 'SMRT Analysis' screen. At the top, there's a breadcrumb 'SMRT Analysis / Analysis Results'. Below it, the job name 'barcoding_tiny_sequel_manual' is displayed. To the right of the job name, the status 'SUCCESSFUL' is shown in green, along with 'Copy' and 'Delete' buttons. The 'Delete' button is circled in red. Below this, there's a table with columns for 'Analysis', 'Analysis ID', and 'Status'. The table contains one row: 'barcoding_tiny_sequel_manual', '75', and 'SUCCESSFUL: 6 tasks finished'.

Starting a job after viewing sequence data

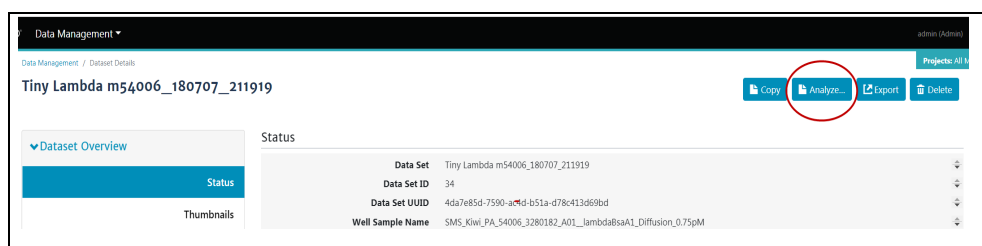
A job can be started by **first** viewing information about specific sequence data:

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of Data Set to use:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 136](#) for details.



4. In the **Name** column, click the name of the sequence data of interest. Details for the selected sequence data display.



5. To **start** a job using this sequence data, click **Analyze**, then follow the instructions starting at Step 12 of [“Creating and starting a job” on page 44](#).

Canceling a running job

1. On the home page, select **SMRT Analysis**.
2. Click the funnel in the **State** column header, then click **Running**. This displays **only** currently-running jobs.
3. Select a currently-running job to cancel.
4. Click **Cancel**.
5. Click **Yes** in the confirmation dialog. The cancelled job displays as **Terminated**.

Restarting a failed job

You can **restart** a failed job; the execution speed from the start to the original point of failure is very fast, which can save time and computing resources. The restarted job **may** run to completion, depending on the source of failure.

Note: As the restarted job uses information from the original failed job, do **not** delete the original job results.

If viewing the results page for the failed job: Click **Restart**.

The screenshot shows the PacBio SMRT Analysis web interface. At the top, the user is logged in as 'mbudagyan (Admin)'. The main header shows 'Bsub_HGAP4 assembly' with a red 'FAILED' status and buttons for 'Send Log Files', 'Restart' (circled in red), 'Copy', and 'Delete'. On the left, a sidebar menu includes 'Analysis Overview', 'Status', 'Thumbnails', 'Display All', 'Coverage', 'Alignment to Draft Assembly', 'Preassembly', and 'Data'. The 'Status' section is active, displaying job details for 'Bsub_HGAP4 assembly' (ID: 101448). The status is 'FAILED: Task genomic_consensus.tasks.variantcaller-10 FAILED in 1496.24 sec'. The job was created by 'sdhillon' on 2019-09-04. The application is 'Assembly (HGAP 4)' and the SMRT Link version is '7.1.0.71715'. The inputs are 'Sequel Data: Demux_32 plex_SD (bc1059--bc1059)'. The path is '/pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448'. The error message states: 'Pbsmrtpipe job /pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448 failed with exit code 1. Task genomic_consensus.tasks.variantcaller-10 task genomic_consensus.tasks.variantcaller failed (exit-code 2) after 1496.24 sec Failed Task genomic_consensus.tasks.variantcaller exit code 2 in 20.37 sec (See file "/pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448/tasks/genomic_consensus.task:10/stderr").' The error is extracted from stderr and indicates 'Unidentifiable sequencing chemistry present in dataset. Check if your SMRTanalysis installation is out-of-date. Unidentifiable sequencing chemistry present in dataset. Check if your'.

If **not** viewing the results page for the failed job:

1. On the home page, select **SMRT Analysis**.
2. Click the funnel in the **State** column header, then click **Failed**. This displays **only** failed jobs.
3. Select a failed job to restart.
4. Click **Restart**.

Viewing job results

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. (**Optional**) Click the funnel in the **State** column header, then click **Successful**. This displays **only** successfully-completed jobs.
3. (**Optional**) Use the Search function to search for specific jobs. See ["Appendix B - Data search" on page 136](#) for details.
4. Click the job link of interest.
5. Click **Analysis Overview > Status** to see job information status, including which application/data utility was used for the job, and the inputs used.
6. Click **Analysis Overview > Thumbnails** or **Display All** to view thumbnails of the reports generated for the job. Click the link under a thumbnail to see a larger image.
7. Depending on the application/data utility used for the job, different job-specific reports are available.
 - For mapping applications **only**: Click **Mapping Report > Summary Metrics** to see an overall summary of the mapping data.
 - For information on the reports and data files produced by analysis applications/data utilities, see ["PacBio® secondary analysis applications" on page 54](#) or ["PacBio® data utilities" on page 90](#).
8. To download data files created by SMRT Link: You can use these data files as input for further processing, pass on to collaborators, or upload to public genome sites. Click **Data > File Downloads**, then click

the appropriate file. The file is downloaded according to your browser settings.

9. **(Optional)** Specify prefix(es) used in the names of files generated by the job. Example: **Run Name** can be included in the name of every file generated by the job. Click **Edit Output File Name Prefix**, check the type(s) of information to append to the file names, then click **Save**.
10. To view job log details: Click **Data > SMRT Link Log**.
11. To visualize the secondary analysis results: See [“Visualizing data using IGV” on page 118](#) for details.

Copying and running an existing job

If you run very similar jobs, you can **copy** an existing job, rename it, optionally modify one or more parameters, then run it.

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. **(Optional)** Click the funnel in the **State** column header, then click **Successful**. This displays **only** successfully-completed jobs.
3. **(Optional)** Use the Search function to search for specific jobs. See [“Appendix B - Data search” on page 136](#) for details.
4. Click the job link of interest.
5. Click **Copy** - this creates a copy of the job, named `Copy of <job name>`, using the **same** parameters.
6. Edit the name of the job.
7. Click **Next**.
8. **(Optional)** Edit any other parameters. See [“PacBio® secondary analysis applications” on page 54](#) or [“PacBio® data utilities” on page 90](#) for further details.
9. Click **Start**.

Exporting a job

You can export the entire contents of a job directory, including the input sequence files, as a ZIP file. Afterwards, deleting the job saves room on the SMRT Link server; you can also later reimport the exported job into SMRT Link if necessary.

1. On the home page, select **SMRT Analysis**.
2. Click **Export Job**.
3. **(Optional)** Use the Search function to search for specific analyses. See [“Appendix B - Data search” on page 136](#) for details.
4. Select one or more jobs to export. This exports the entire contents of the job directory.
5. Click **Export Selected**.
6. Select the output directory for the job data and click **Export**.

Importing a job

Note: You can **only** import a job that was created in SMRT Link, then exported.

1. On the home page, select **SMRT Analysis**.

2. Click **Import Job**.
3. Select a ZIP file containing the job to import.
4. Click **Import**. The job is imported and is available on the main SMRT Analysis page.

PacBio® secondary analysis applications

Following are the secondary analysis applications provided with SMRT Analysis v11.0. These applications are designed to produce biologically-meaningful results. Each application is described later in this document, including all analysis parameters, reports and output files generated by the application.

Note: These applications accept **only** HiFi reads as input.

Genome Assembly

- Generate *de novo* assemblies of genomes, using HiFi reads.
- See [“Genome Assembly” on page 55](#) for details.

HiFi Mapping (was Mapping)

- Align (or map) reads to a user-provided reference sequence.
- See [“HiFi Mapping” on page 58](#) for details.

HiFiViral SARS-CoV-2 Analysis

- Analyze multiplexed viral surveillance samples for SARS-CoV-2, using HiFi reads.
- See [“HiFiViral SARS-CoV-2 Analysis” on page 62](#) for details.

Iso-Seq® Analysis

- Characterize full-length transcript isoforms, using HiFi reads.
- See [“Iso-Seq® Analysis” on page 67](#) for details.

Microbial Genome Analysis

- **Note:** This combines and replaces the **Microbial Assembly** and **Base Modification Analysis** applications in the previous release.
- Generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb, and identify methylated bases and associated nucleotide motifs.
- Optionally include identification of 6mA and 4mC modified bases and associated DNA sequence motifs.
- See [“Microbial Genome Analysis” on page 74](#) for details.

Minor Variants Analysis

- Identify and phase minor single nucleotide substitution variants in complex populations.
- See [“Minor Variants Analysis” on page 80](#) for details.

Structural Variant Calling

- Identify structural variants (Default: ≥ 20 bp) in a sample or set of samples relative to a reference.
- See [“Structural Variant Calling” on page 86](#) for details.

Genome Assembly

Use this application to generate high quality *de novo* assemblies of genomes, using HiFi reads.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

The application includes seven main steps:

1. Convert input to a compressed database for fast retrieval.
2. Overlap reads using the **Pancake** tool.
3. Phase the overlapped reads using **Nighthawk**. Nighthawk also boosts contiguity of the assembly by removing overlaps between reads coming from different instances of a genomic repeat (such as segmental duplications.)
4. Remove chimeras and duplicate reads which do not span repeat regions. This improves contiguity and assembly quality.
5. Construct a string graph. Extract primary contigs and haplotigs. Haplotypes are represented by heterozygous bubbles.
6. Polish the contigs and haplotigs using phased reads. Phasing information is preserved. Polishing is done with [Racon](#).
7. Identify potential haplotype duplications in the primary contig set using the [purge_dups](#) tool, and move them to the haplotig set. This final round of assembly processing is especially useful in high heterozygosity samples.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Parameters

Advanced parameters	Default value	Description
Genome Length	0	The approximate number of base pairs expected in the genome. This is used only for downsampling; if the value is ≤ 0 , downsampling is disabled. Enter an integer, optionally followed by one of the metric suffixes: k, M or G. Example: 4500k means "4,500 kilobases" or "4,500,000". M stands for Mega and G stands for Giga.
Downsampled coverage	0	The input Data Set can be downsampled to a desired coverage, provided that both the Downsampled Coverage and Genome Length parameters are specified and > 0 . Downsampling applies to the entire assembly process, including polishing. This parameter selects reads randomly, using a fixed random seed for reproducibility.

Advanced parameters	Default value	Description
Run polishing	ON	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies.
Run phasing	ON	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Advanced Assembly Options	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted.
Purge duplicate contigs from the assembly	ON	Enables or disables identification of "duplicate" alternate haplotype contigs which may be assembled in the primary contig file, and moves them to the associate contig (haplotig) file.
Cleanup intermediate Files	ON	Removes intermediate files from the run directory to save space.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Genome Assembly application generates the following reports:

Polished Assembly > Summary Metrics

Displays statistics on the contigs from the *de novo* assembly that were corrected by Racon.

- **Contig Type:** Primary or Haplotigs. Primary contigs represent pseudohaplotype assemblies, while haplotigs represent fully phased and assembled regions of the genome. Primary contigs are usually much longer than haplotigs due to allowed haplotype switching.
- **Polished Contigs:** The number of polished contigs.
- **Maximum Contig Length:** The length of the longest contig.
- **Mean Contig Length:** The mean length of the contigs.
- **Median Contig Length:** The median length of the contigs.
- **N50 Contig Length:** 50% of the contigs are longer than this value.
- **Sum of Contig Lengths:** The total length of all the contigs.
- **E-size (sum of squares/sum):** The expected contig size for a random base in the polished contigs. Another interpretation: The area under the Nx curve (for x in range [0, 100]).
- **Number of Circular Contigs:** The number of assembled contigs that are circular.

Polished Assembly > Polished Contigs

- **Contig:** The name of the individual contig.
- **Length (bases):** The length of the contig, in bases.
- **Circular:** **Yes** if the contig is circular, **No** if it isn't.
- **Percent Polished:** The percent of contig bases that were polished.

- **Number of Polishing Reads:** The number of reads used to perform polishing on this contig.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Haplotigs:** The final polished haplotigs assembly, in FASTA format.
- **Primary Contigs:** The final polished primary contigs assembly, in FASTA format.

HiFi Mapping

Use this application to align (or map) data to a user-provided reference sequence. The HiFi Mapping application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Maps data to a provided reference sequence, and then identifies consensus and variants against this reference.
- Haploid variants and small indels, but **not** diploid variants, are called as a result to alignment to the reference sequence.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Set (Required)

- Specify a reference sequence to align the SMRT Cells reads to and to produce alignments.

Consolidate Mapped BAMs for IGV (Default = OFF)

- By default, SMRT Link consolidates chunked BAM files for viewing in IGV if the combined size is not more than 10 GB. Setting this option to **ON ignores** the file size cutoff and consolidates the BAM files.
- **Note:** This setting can **double** the amount of storage used by the BAM files, which can be considerable. Make sure to have enough disk space available. This setting may also result in longer run times.

Parameters

Advanced parameters	Default value	Description
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Minimum Mapped Length (bp)	50	The minimum required mapped read length, in base pairs.
Bio Sample Name of Aligned Dataset	NONE	Populates the Bio Sample Name (Read Group <small>SM</small> tag) in the aligned BAM file. If blank, uses the Bio Sample Name of the input file. Note: Avoid using spaces in Bio Sample Names as this may lead to third-party compatibility issues.
Minimum Gap-Compressed Identity (%)	70	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.

Advanced parameters	Default value	Description
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.0 for details.
Target Regions (BED file)	NONE	(Optional) Specifies a BED file that defines regions for a Target Regions report showing coverage over those regions. See “Appendix C - BED file format for Target Regions report” on page 138 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The HiFi Mapping application generates the following reports:

Target Regions > Target Regions

Displays the number (and percentage) of reads that hit target regions specified by an input BED file. This is useful for targeted DNA sequencing applications. (This report displays **only** if a BED file is specified when creating the analysis.)

- **Coordinates:** The chromosome coordinates, as specified in the input BED file.
- **Region:** The name of the region, as specified in the input BED file.
- **On-Target Reads:** The number (and percentage) of unique reads that map with any overlap to the target region.

Target Regions > Target Region Coverage

- Displays the number of hits per defined region of the chromosome.

Mapping Report > Summary Metrics

Mapping is local alignment of a read or subread to a reference sequence.

- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).
- **Number of Alignments:** The number of alignments that mapped to the reference sequence.
- **Number of CCS reads (total):** The total number of CCS reads in the sequence.
- **Number of CCS reads (mapped):** The number of CCS reads that mapped to the reference sequence.
- **Number of CCS reads (unmapped):** The number of CCS reads not mapped to the reference sequence.
- **Percentage of CCS reads (mapped):** The percentage of CCS reads that mapped to the reference sequence.
- **Percentage of CCS reads (unmapped):** The percentage of CCS reads not mapped to the reference sequence.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.

- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **CCS Read Length 95% (mapped):** The 95th percentile of read length of CCS reads that mapped to the reference sequence.
- **CCS Read Length Max (mapped):** The maximum length of CCS reads that mapped to the reference sequence.

Mapping Report > CCS Mapping Statistics Summary

Displays mapping statistics per movie.

- **Sample:** The sample name for which the following metrics apply.
- **Movie:** The movie name for which the following metrics apply.
- **Number of CCS Reads (mapped):** The number of CCS reads that mapped to the reference sequence. This includes adapters.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read Length N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped CCS Read Length

- Histogram distribution of the number of mapped CCS reads by read length.

Mapping Report > Mapped CCS Reads Concordance

- Histogram distribution of the number of CCS reads by the percent concordance with the reference sequence. Concordance for CCS reads is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped Concordance vs Alignment Length

- Maps the percent concordance with the reference sequence against the alignment length, in base pairs.

Coverage > Summary Metrics

- **Mean Coverage:** The mean depth of coverage across the reference sequence.
- **Missing Bases:** The percentage of the reference sequence without coverage.

Coverage > Coverage Across Reference

- Maps coverage across the reference.

Coverage > Depth of Coverage

- Maps the reference regions against the percent coverage.

Coverage > Coverage vs. [GC] Content

- Maps (as a percentage, over a 100 bp window) the number of Gs and Cs present across the coverage. The number of genomic windows with the corresponding % of Gs and Cs is displayed on top. Used to check that no coverage is lost over extremely biased base compositions.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Mapped Reads:** All input reads that were mapped to the reference by the application.
- **Coverage Summary:** Coverage summary for regions (bins) spanning the reference sequence.
- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see [“Visualizing data using IGV” on page 118](#) for details.

- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.

**HiFiViral
SARS-CoV-2
Analysis**

Use this application to analyze multiplexed samples sequenced with the HiFiViral SARS-CoV-2 kit. For **each** sample, this analysis provides:

- Consensus sequence (FASTA).
- Variant calls (VCF).
- HiFi reads aligned to the reference (BAM).
- Plot of HiFi read coverage depth across the SARS-CoV-2 genome.

Across **all** samples, this analysis provides:

- Job summary table including passing sample count at 90 and 95% genome coverage.
- Sample summary table including, for each sample: Count of variable sites, genome coverage, read coverage, and probability of multiple strains, and other metrics.
- Plate QC graphical summary of performance across samples in assay plate layout.
- Plot of HiFi read depth of coverage for all samples.

Notes:

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Phred-scaled Q20.
- This application is for SARS-CoV-2 analysis **only** and is **not** recommended for other viral studies. The Wuhan reference genome is provided by default to run the application, but advanced users may specify other reference genomes. We have **not** tested the application with reference genomes other than the Wuhan reference genome.
- The application is intended to identify variable sites and call a single consensus sequence per sample. The output consensus sequence is produced based on the dominant variant observed. Minor variant information that passes through a default threshold may be encoded in the raw VCF, but does **not** get propagated into the consensus sequence FASTA.
- The HiFiViral SARS-CoV-2 Analysis application can be run using the **Auto Analysis** feature available in Run Design. This feature allows users to complete all necessary analysis steps immediately after sequencing **without** manual intervention. The Auto Analysis workflow includes CCS, Demultiplex Barcodes, and HiFiViral SARS-CoV-2 Analysis.

Auto Analysis in Run Design

Users may set the analysis to begin **automatically** after sequencing completes using Auto Analysis in Run Design. See [“HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design” on page 116](#) for details.

HiFiViral SARS-CoV-2 application workflow

1. Process the reads using the `mimux` tool to trim the probe arm sequences.
2. Align the reads to the reference genome using `pbbmm2`.
3. Call and filter variants using `bcftools`, generating the raw variant calls in VCF file format. Filtering in this step removes low-quality calls (less than Q20), and normalizes indels.
4. Filter low-frequency variants using `vcfcons` and generate a consensus sequence by injecting variants into the reference genome. At each position, a variant is called **only** if **both** the base coverage **exceeds** the minimum base coverage threshold (Default = 4) **and** the fraction of reads that support this variant is **above** the minimum variant frequency threshold (Default = 0.5). See [here](#) for details.

Preparing input data for the HiFiViral SARS-CoV-2 Analysis application

1. Run the **Demultiplex Barcodes** data utility, where the inputs are HiFi reads, and the primers are multiplexed barcode primers. (If HiFi reads have **not** been generated on the instrument, run CCS analysis first. See [“Circular Consensus Sequencing \(CCS\)” on page 104](#) for details.)
 - The proper barcode sequences are provided by default:
`Barcoded M13 Primer Plate.`
 - For the **Same Barcodes on Both Ends of Sequence** parameter, specify **No**; the barcode pairs are **asymmetric**.
 - Provide the correctly-formatted barcode pair-to-Bio Sample CSV file for the **Assign Bio Sample Names to Barcodes** option. (For details, see [“Assign Bio Sample Names to Barcodes \(Required\)” on page 93.](#))

Running the HiFiViral SARS-CoV-2 Analysis application

1. **After** running the Demultiplex Barcode data utility, create a new job using **SMRT Analysis > Create New Job**.
2. Name the job.
3. Select all the demultiplex samples contained in the Data Set and choose **Analysis of Multiple Data Sets > One Analysis for All Data Sets**. Click **Next**.
4. Select **HiFiViral SARS-CoV-2 Analysis** from the Analysis Application list.
5. **SARS-CoV-2 Genome NC_045512.2** (the Wuhan reference genome) is automatically loaded; advanced users may select a different reference if desired.
6. To generate the optional Plate QC graphical summary, click **Advanced Parameters** and load a CSV file using the provided template (**assay-PlateQC_template_4by96.csv**) as a guide.
7. Click **OK**, then **Start**.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.

- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Genome (Required)

- Specify the full viral genome against which to align the reads and call variants. (The default is the Wuhan Reference genome.)

Parameters

Advanced parameters	Default value	Description
Plate QC CSV	NONE	(Optional) Specify a CSV file to generate the Plate QC report, which displays analysis results for each sample in the assay plate. The CSV file must contain barcode (asymmetric pairs), Bio Sample Name, assay plate IDs (can include 1-4 plates with unique names; avoid special characters), and assay plate well IDs in the format A01, A02, ...H12. (To create a new file, click Download Template , edit, and then save the CSV file.) The plate and well information corresponds to the location of samples during the SARS-CoV-2 enrichment assay.
Probes FASTA	NONE	Specify probe sequences in FASTA format if using probes other than the standard probes shipped in the HiFiViral SARS-CoV-2 Kit.
Minimum Base Coverage	4	Specify the minimum read depth at each position to report either a variant or a reference base. Positions with less than this specified coverage will have an N base output in the consensus sequence FASTA file. Increasing the minimum base coverage may result in more Ns and loss of variant detection. We do not recommend making this value lower than the default threshold of 4, as it may increase the number of false positive variants called.
Minimum Variant Frequency	0.5	Specify that only variants whose frequency is greater than this value are reported. This frequency is determined based on the read depth (DP) and allele read count (AD) information in the VCF output file. We recommend using the default value to properly call the dominant alternative variant while also filtering out potential artifacts.
Advanced Processing Options	NONE	Additional options to pass to the <code>mimux</code> preprocessing tool for trimming and filtering reads by probe sequences. Options should be entered in space-separated format. See the HiFiViral SARS-CoV-2 Analysis section of SMRT Tools reference guide (v11.0) for details.
Minimum Barcode Score	80	A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator of how well the chosen barcode pair matches. It ranges between 0 (no match) and 100 (a perfect match). This parameter specifies that reads with barcode scores below this minimum value are not included in analysis.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The HiFiViral SARS-CoV-2 Analysis application generates the following reports:

Summary Report > Summary Metrics

- **Samples:** The count of all input samples, whether or not they passed analysis.
- **Samples with Genome Coverage > 90%:** The number of samples where at least 90% of bases have at least four mapped reads overlapping their position.
- **Samples with Genome Coverage > 95%:** The number of samples where at least 95% of bases have at least four mapped reads overlapping their position.
- **Samples Failing Workflow:** The number of samples for which the analysis was unable to generate a per-sample report due to an absence of usable data.

Summary Report > Sample Summary

- **Bio Sample Name:** The name of the biological sample associated with the variants. (**Note:** Any spaces in the name are substituted by new line characters for consistency with output file names.)
- **Substitutions:** The count of all called substitutions in the consensus sequence for the sample.
- **Insertions:** The count of all called insertions in the consensus sequence for the sample.
- **Deletions:** The count of all called deletions in the consensus sequence for the sample.
- **Reads:** The total number of HiFi reads for the sample.
- **Read Coverage:** The mean number of mapped reads overlapping with each position in the reference genome.
- **On-Target Rate:** The mapping yield of reads; the number of unique mapped reads divided by the total number of reads.
- **Multiple Strains (Probability):** Samples are flagged as having multiple strains if the probability is at least 0.95. Samples may contain multiple strains due to sample contamination or presence of multiple strains in the RNA extract. To classify a sample as multi-strain, we tolerate error by using the binomial cumulative distribution function (with a fixed probability of 0.2). This feature is supported for samples with Ct < 26 with minor frequencies > 20%. Samples **must** have > 70% genome coverage to be called Multiple Strains.
- **Ns:** The number of bases in the consensus sequence that are Ns.
- **Genome Coverage:** The percentage of bases with at least four mapped reads overlapping their position by default. See the **Advanced Parameters** dialog to adjust minimum base coverage.

Summary Report > Genome Coverage

- Coverage plot showing the per-sample mean read coverage within a window of 100 bp. The shaded region displays the 25th to 75th percentile in the range of coverage across all samples, and the darker solid line displays the **median** coverage across all samples.

Summary Report > Plate QC

Plot showing analysis results for each plate cell used. This plot is generated **only** if the user supplies a Plate QC CSV file mapping Bio Sample Names to Well IDs in **Advanced Parameters**.

- **Blue** wells represent samples with at least 95% coverage.
- **Green** wells represent samples with at least 90% coverage.
- **Yellow** wells represent samples that passed the workflow but had genome coverage worse than 90%.
- **Red** wells represent samples that failed the workflow.

- **White** wells do **not** include a sample.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **All Samples, HiFi Reads FASTQ:** HiFi reads in FASTQ format for all samples.
- **All Samples, Consensus Sequence FASTA:** The full consensus genomic sequences; bases for which **no** consensus could be called are represented by **Ns**. See the **Advanced Parameters** dialog to adjust the minimum base coverage for outputting **Ns**.
- **All Samples, Genome Coverage Plots:** Plots for individual samples showing coverage depth across the genome.
- **All Samples, Variant Call VCF:** VCF file containing the final variant calls per sample.
- **All Samples, HiFi Reads Mapped BAM:** BAM file for each sample containing the HiFi reads aligned to the reference genome.
- **All Samples, Consensus Sequence Aligned BAM:** BAM file for each sample of consensus sequence aligned to the reference genome. The consensus sequence is split into fragments where there are **Ns** and each fragment is mapped.
- **All Samples, Raw Variant Calls VCF:** VCF file containing the intermediate variant calls per patient sample.
- **Sample Summary Table CSV:** CSV version of the data shown in the Sample Summary table.
- **All Samples, Probe Counts TSV:** Tab-delimited text file containing per-sample, per-probe counts. This file can be used to identify samples that are poorly sequenced or probes with high or low coverages.
- **Sample Inputs CSV:** CSV version of the Plate QC CSV, if supplied in the **Advanced Parameters** dialog.

**Iso-Seq®
Analysis**

The Iso-Seq application enables analysis and functional characterization of full-length transcript isoforms for sequencing data generated on PacBio instruments.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Q20.

Notes on Multiplexed Data

There are two ways in which an Iso-Seq library can be multiplexed:

1. Barcoded adapter Iso-Seq libraries

- If using the SMRTbell Barcoded Adapter with the Iso-Seq Express protocol **on or after** April 21, 2022, demultiplex the Data Set **prior** to running the Iso-Seq application.
- To analyze the samples in a single Iso-Seq run, select **all** the demultiplexed Data Sets to combine and begin the Iso-Seq analysis.
- If following the standard Iso-Seq Express protocol, select **Iso-Seq cDNA Primers** as the Primer Set.

2. Barcoded cDNA primer Iso-Seq libraries

- If following multiplexing guidelines using the Iso-Seq Express protocol **on or prior to** April 21, 2022 and you ordered synthesized oligos listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document **Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell prep kit 3.0**, demultiplex your Data Set using the Iso-Seq application. In other words, do **not** run the Demultiplexing Barcodes utility first.
- See the **Primer Set Selection** column below for the correct choice of primer sequences.

Multiplexed method	Demultiplexed before Iso-Seq?	Primer set selection
Not multiplexed	NO	Iso-Seq cDNA Primers
Barcoded adapters	YES	Iso-Seq cDNA Primers
Barcoded cDNA primer	NO	Iso-Seq 12 Barcoded cDNA Primers Or Custom cDNA Primers

The application includes three main steps:

1. **Classify:** Identify and remove primers (which may be cDNA primers or barcoded cDNA primers). Identify full-length reads based on the asymmetry of 5' and 3' primers. Trim off polyA tails and remove artifactual concatemers.
2. **Cluster (Optional):** Perform *de novo* clustering and consensus calling. Output full-length consensus isoforms that are further separated into high-quality (HQ) and low-quality (LQ) based on estimated accuracies.

3. **Collapse (Optional):** When a reference genome is selected, map HQ isoforms to the genome, then collapse redundant isoforms into unique isoforms.

To obtain full-length non-concatemer (FLNC) reads and **not** complete the Cluster step: Ensure that the **Run Clustering** option is set to **OFF**.

Iso-Seq determines two FLNC reads to be the same isoform, and will place them in the same cluster, if the two reads:

- Differ less than 100 bp on the 5' end.
- Differ less than 30 bp on the 3' end.
- Have no internal gaps that exceed 10 bp.

Iso-Seq will **only** output clusters that have at least two FLNC reads.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Primer Set (Required)

- Specify a primer sequence file in FASTA format to identify cDNA primers for removal. The primer sequence includes the 5' and 3' cDNA primers and (if applicable) barcodes.
- Primer IDs **must** be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should **not** include the Ts and is written in reverse complement (see examples below).
- Each primer sequence must be **unique**.

Example 1: The Iso-Seq cDNA Primer primer set, included with the SMRT Link installation.

Users following the standard Iso-Seq Express protocol **without** multiplexing, or running a Data Set that has **already** been demultiplexed (either using Run Design or the SMRT Analysis application) should use this default option.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>IsoSeq_3p
GTACTCTGCGTTGATACCACTGCTT
```

Example 2: The Iso-Seq 12 Barcoded cDNA Primers set, included with the SMRT Link installation.

Users using barcoded cDNA primers listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document **Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell prep kit 3.0**, should select this option.

```
>bc1001_5p
CACATATCAGAGTGC GGCAATGAAGTCGCAGGGTTGGG
>bc1002_5p
ACACACAGACTGTGAGGCAATGAAGTCGCAGGGTTGGG
...
```

(There are a total of 24 sequence records, representing 12 pairs of F/R barcoded cDNA primers.)

Example 3: An example of a custom cDNA primer set. 4 tissues were multiplexed using barcodes on the 3' end only.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>dT_BC1001_3p
AAGCAGTGGTATCAACGCAGAGTACCACATATCAGAGTGCG
>dT_BC1002_3p
AAGCAGTGGTATCAACGCAGAGTACACACACAGACTGTGAG
>dT_BC1003_3p
AAGCAGTGGTATCAACGCAGAGTACACACATCTCGTGAGAG
>dT_BC1004_3p
AAGCAGTGGTATCAACGCAGAGTACCACGCACACACGCGCG
```

Example 4: Special Handling for the TeloPrime cDNA Kit

The Lexogen TeloPrime cDNA kit contains **As** in the 3' primer that **cannot** be differentiated from the polyA tail. For best results, remove the **As** from the 3' end as shown below:

```
>TeloPrimeModified_5p
TGGATTGATATGTAATACGACTCACTATAG
>TeloPrimeModified_3p
CGCCTGAGA
```

Reference Set (Optional)

- Optionally specify a reference genome to align High Quality isoforms to, and to collapse isoforms mapped to the same genomic loci.

Run Clustering (Default = ON)

- Specify **ON** to generate consensus isoforms.
- Specify **OFF** to classify reads **only** and not generate consensus isoforms. The Reference Set will also be ignored.

Cluster Barcoded Samples Separately (Default = OFF)

- Specify **OFF** if barcoded samples are from the **same** species, but different tissues, or samples of the same genes but different individuals. The samples are clustered with **all** barcodes pooled.
- Specify **ON** if barcoded samples are from **different** species. The samples are clustered separately by barcode.
- In either case, the samples on the results page are automatically named `BioSample_1` through `BioSample_N`.

Parameters

Advanced parameters	Default value	Description
Require and trim Poly(A) Tail	ON	ON means that polyA tails are required for a sequence to be considered full length. OFF means sequences do not need polyA tails to be considered full length.
Minimum Mapped Length (bp)	50	The minimum required mapped HQ isoform sequence length (in base pairs) for the Iso-Seq mapping-collapse step. Note: This is applicable only if a reference genome is provided.
Minimum Gap-Compressed Identity (%)	95	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference. Note: This is applicable only if a reference genome is provided.
Minimum Mapped Coverage (%)	99	The minimum required HQ transcript isoform sequence alignment coverage (in percent) for the Iso-Seq mapping-collapse step. Note: This is applicable only if a reference genome is provided.
Maximum Fuzzy Junction Difference (bp)	5	The maximum junction difference between two mapped isoforms to be collapsed into a single isoform. If the junction differences are all less than the provided value, they will all be collapsed. Setting to 0 requires all junctions to be exact to be collapsed into a single isoform. Applicable only if a reference genome is provided.
Min. CCS Predicted Accuracy (Phred Scale)	10	Phred-scale integer QV cutoff for filtering HiFi reads. The default for Iso-Seq Analysis is 20 (QV 20), or 99% predicted accuracy.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. (pbmm2 is already running with <code>--preset ISOSEQ</code> .) Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.0 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Iso-Seq application generates the following reports:

CCS Analysis Read Classification > Summary Metrics

- **Reads:** The total number of CCS reads.
- **Reads with 5' and 3' Primers:** The number of CCS reads with 5' and 3' cDNA primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers:** The number of non-concatemer CCS reads with 5' and 3' primers detected.

- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail:** The number of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected. This is usually the number for full-length, non-concatemer (FLNC) reads, unless polyA tails are not present in the sample.
- **Mean Length of Full-Length Non-Concatemer Reads:** The mean length of the non-concatemer CCS reads with 5' and 3' primers and polyA tails detected.
- **Unique Primers:** The number of unique primers in the sequence.
- **Mean Reads per Primer:** The mean number of CCS reads per primer.
- **Max. Reads per Primer:** The maximum number of CCS reads per primer.
- **Min. Reads per Primer:** The minimum number of CCS reads per primer.
- **Reads without Primers:** The number of CCS reads without a primer.
- **Percent Bases in Reads with Primers:** The percentage of bases in CCS reads in the sequence data that contain primers.
- **Percent Reads with Primers:** The percentage of CCS reads in the sequence data that contain primers.

CCS Analysis Read Classification > Primer Data

- **Bio Sample Name:** The name of the biological sample associated with the primer.
- **Primer Name:** A string containing the pair of primer indices associated with this biological sample.
- **CCS Reads:** The number of CCS reads associated with the primer.
- **Mean Primer Quality:** The mean primer quality associated with the primer.
- **Reads with 5' and 3' Primers:** The number of CCS reads with 5' and 3' cDNA primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers:** The number of non-concatemer CCS reads with 5' and 3' primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail:** The number of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected. This is usually the number for full-length, non-concatemer (FLNC) reads, unless polyA tails are not present in the sample.

CCS Analysis Read Classification > Primer Read Statistics

- **Number Of Reads Per Primer:** Maps the number of reads per primer, sorted by primer ranking.
- **Primer Frequency Distribution:** Maps the number of samples with primers by the number of reads with primers.
- **Mean Read Length Distribution:** Maps the read mean length against the number of samples with primers.

CCS Analysis Read Classification > Primer Quality Scores

- Histogram of primer scores.

CCS Analysis Read Classification > Length of Full-Length Non-Concatemer Reads

- Histogram of the read length distribution of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected.

Transcript Clustering > Summary Metrics

- **Sample Name:** The sample name for which the following metrics apply.
- **Number of High-Quality Isoforms:** The number of consensus isoforms that have an estimated accuracy **above** the specified threshold.
- **Number of Low-Quality Isoforms:** The number of consensus isoforms that have an estimated accuracy **below** the specified threshold.

Transcript Clustering > Length of Consensus Isoforms

- Histogram of the consensus isoform lengths and the distribution of isoforms exceeding a read length cutoff.

Transcript Mapping > Summary Metrics

- **Sample Name:** Sample name for which the following metrics apply.
- **Number of mapped unique isoforms:** The number of unique isoforms, where each unique isoform is generated by collapsing redundant HQ isoforms (such as those have very minor differences from one to one another) to one isoform. Each unique isoform may be generated from one or multiple HQ isoforms.
- **Number of mapped unique loci:** The number of unique mapped genomic loci among all unique isoforms. Multiple unique isoforms may map to the same genomic location, indicating these unique isoforms are transcribed from the same gene family, but spliced differently.

Transcript Mapping > Length of Mapped Isoforms

- Histogram of mapped isoforms binned by read length and the distribution of mapped isoforms exceeding a read length cutoff.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Primers Summary:** Text file listing how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- **Inferred Primers:** Inferred primers used in the analysis. The algorithm looks at the first 35,000 ZMWs, then selects primers with ≥ 10 counts and mean scores ≥ 45 .
- **Full-Length Non-Concatemer Read Assignments:** Full-length reads that have primers and polyA tails removed, in BAM format.
- **Full-Length Non-Concatemer Report:** Includes strand, 5' primer length, 3' primer length, polyA tail length, insertion length, and primer IDs for each full-length read that has primers and polyA tail, in CSV format.
- **Low-Quality Isoforms:** Isoforms with low consensus accuracy, in FASTQ and FASTA format. We recommend that you work only with High-Quality isoforms, unless there are specific reasons to analyze Low-Quality isoforms. When the input Data Set is a ConsensusReadSet, a FASTA file **only** is generated.
- **High-Quality Isoforms:** Isoforms with high consensus accuracy, in FASTQ and FASTA format. This is the recommended output file to work with. When the input Data Set is a ConsensusReadSet, a FASTA file **only** is generated.
- **Cluster Report:** Report of each full-length read into isoform clusters.
- **Isoform Counts by Barcode:** For each isoform, report supportive FLNC reads for each barcode.
- **Mapped High Quality Isoforms:** Alignments mapping isoforms to the reference genome, in BAM and BAI (index) formats.
- **Collapsed Filtered Isoforms GFF:** Mapped, unique isoforms, in GFF format. This is the Mapping step output that is the recommended output file to work with.

- **Collapsed Filtered Isoforms:** Mapped, unique isoforms, in FASTQ format. This is the Mapping step output that is recommended output file to work with. When the input Data Set is a ConsensusReadSet, **only** a FASTA file is generated.
- **Collapsed Filtered Isoforms Groups:** Report of isoforms mapped into collapsed filtered isoforms.
- **Full-length Non-Concatemer Read Assignments:** Report of full-length read association with collapsed filtered isoforms, in text format.
- **Collapsed Filtered Isoform Counts:** Report of read count information for each collapsed filtered isoform.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see [“Visualizing data using IGV” on page 118](#) for details.

- **Mapped High Quality Isoforms:** Alignments mapping isoforms to the reference genome, in BAM and BAI (index) formats.

Note: For details on custom PacBio tags added to output BAM files by the Iso-Seq Application, see page 54 of **SMRT Tools reference guide (v11.0)**, or see [here](#) for details.

**Microbial
Genome
Analysis**

Use this application to generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb. This application can optionally include analysis of 6mA and 4mC modified bases and associated DNA sequence motifs. (This requires kinetic information.)

Note: This combines and replaces the **Microbial Assembly** and **Base Modification Analysis** applications in the previous release.

The Microbial Genome Analysis application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Includes chromosomal- and plasmid-level *de novo* genome assembly, circularization, polishing, and rotation of the origin of replication for each circular contig.
- Performs base modification detection to identify 4mCm and 6mA and associated DNA sequence motifs. (This requires kinetic information.)
- Facilitates assembly of larger genomes (yeast) as well.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Run Base Modification Analysis (Default = ON)

- Run Base Modification analysis on the final assembly. This **only** applies if the assembly is not empty, and the input data contains the correct kinetic tags.

Find Modified Base Motifs (Default = ON)

- Perform motif detection on the results of base modification analysis.

Parameters

Advanced parameters	Default value	Description
Advanced Assembly Options for chromosomal stage	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. See Appendix C in SMRT Tools reference guide (v11.0) for details.
Advanced Assembly Options for plasmid stage	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. See Appendix C in SMRT Tools reference guide (v11.0) for details.
Maximum plasmid length, bp	300,000	Value that should be set higher than the maximum size of a plasmid in the input sample. The default value should work well in most cases.

Advanced parameters	Default value	Description
Run secondary polish	ON	Specify that an additional polishing stage be run at the end of the workflow.
Base modifications to identify	m4C,m6A	Specify the base modifications to identify, in a comma-separated list.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Cleanup intermediate files	ON	Removes intermediate files from the run directory to save space.
Minimum Qmod Score	35	Specify the minimum Qmod score to use in motif-finding.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Microbial Genome Analysis application generates the following reports:

Mapping Report > Summary Metrics

Mapping is local alignment of a read or subread to a reference sequence.

- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).
- **Number of Alignments:** The number of alignments that mapped to the reference sequence.
- **Number of CCS reads (total):** The total number of CCS reads in the sequence.
- **Number of CCS reads (mapped):** The number of CCS reads that mapped to the reference sequence.
- **Number of CCS reads (unmapped):** The number of CCS reads not mapped to the reference sequence.
- **Percentage of CCS reads (mapped):** The percentage of CCS reads that mapped to the reference sequence.
- **Percentage of CCS reads (unmapped):** The percentage of CCS reads not mapped to the reference sequence.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **CCS Read Length 95% (mapped):** The 95th percentile of read length of CCS reads that mapped to the reference sequence.
- **CCS Read Length Max (mapped):** The maximum length of CCS reads that mapped to the reference sequence.

Mapping Report > CCS Mapping Statistics Summary

Displays mapping statistics per movie.

- **Sample:** The sample name for which the following metrics apply.
- **Movie:** The movie name for which the following metrics apply.
- **Number of CCS Reads (mapped):** The number of CCS reads that mapped to the reference sequence. This includes adapters.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read Length N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped CCS Read Length

- Histogram distribution of the number of mapped CCS reads by read length.

Mapping Report > Mapped CCS Reads Concordance

- Histogram distribution of the number of CCS reads by the percent concordance with the reference sequence. Concordance for CCS reads is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped Concordance vs Read Length

- Maps the percent concordance with the reference sequence against the CCS read length, in base pairs.

Polished Assembly > Summary Metrics

Displays statistics on the contigs from the *de novo* assembly that were corrected by Arrow.

- **Polished Contigs:** The number of polished contigs.
- **Maximum Contig Length:** The length of the longest contig.
- **N50 Contig Length:** 50% of the contigs are longer than this value.
- **Sum of Contig Lengths:** Total length of all the contigs.
- **E-size (sum of squares/sum):** The expected contig size for a random base in the polished contigs.

Polished Assembly > Polished Contigs from Microbial Assembly HiFi

Displays a table of details about all assembled contigs.

- **Contig:** The contig name.
- **Length:** The length of the contig, in base pairs, after polishing.

- **Circular:** Marks whether circularity of the contig was detected. Output values are `yes` and `no`.
- **Coverage:** The average coverage across the contig, calculated by the sum of coverage of all bases in the contig divided by the number of bases.

Coverage > Summary Metrics

Displays depth of coverage across the *de novo*-assembled genome, as well as depth of coverage distribution.

- **Mean Coverage:** The mean depth of coverage across the assembled genome sequence.
- **Missing Bases:** The percentage of the genome's sequence that have zero depth of coverage.

Coverage > Coverage across Reference

- Displays coverage at each position of the draft genome assembly.

Coverage > Depth of Coverage

- Histogram distribution of the draft assembly regions by the coverage.

Coverage > Coverage vs. [GC] Content

- Maps (as a percentage, over a 100 bp window) the number of Gs and Cs present across the coverage. The number of genomic windows with the corresponding % of Gs and Cs is displayed on top. Used to check that no coverage is lost over extremely biased base compositions.

Base Modifications > Kinetic Detections

- **Per-Base Kinetic Detections:** Maps the modification QV against per-strand coverage.
- **Kinetic Detections Histogram:** Histogram distribution of the number of bases by modification QV.

Modified Base Motifs > Modified Base Motifs

Displays statistics for the methyltransferase recognition motifs detected.

- **Motif:** The nucleotide sequence of the methyltransferase recognition motif, using the standard IUPAC nucleotide alphabet.
- **Modified Position:** The position within the motif that is modified. The first base is 0. **Example:** The modified adenine in GATC is at position 2.
- **Modification Type:** The type of chemical modification most commonly identified at that motif. These are: `6mA`, `4mC`, or `modified_base` (modification not recognized by the software.)
- **% of Motifs Detected:** The percentage of times that this motif was detected as modified across the entire genome.
- **# of Motifs Detected:** The number of times that this motif was detected as modified across the entire genome.
- **# of Motifs In Genome:** The number of times this motif occurs in the genome.
- **Mean QV:** The mean modification QV for all instances where this motif was detected as modified.
- **Mean Coverage:** The mean coverage for all instances where this motif was detected as modified.

- **Partner Motif:** For motifs that are not self-palindromic, this is the complementary sequence.
- **Mean IPD Ratio:** The mean inter-pulse duration. An IPD ratio greater than 1 means that the sequencing polymerase slowed down at this base position, relative to the control. An IPD ratio less than 1 indicates speeding up.
- **Group Tag:** The motif group of which the motif is a member. Motifs are grouped if they are mutually or self reverse-complementary. If the motif isn't complementary to itself or another motif, the motif is given its own group.
- **Objective Score:** For a given motif, the objective score is defined as $(\text{fraction methylated}) * (\text{sum of log-p values of matches})$.

Modified Base Motifs > Modification QVs

- Maps motif sites against Modification QV for all genomic occurrences of a motif, for each reported motif, including "No Motif".

Modified Base Motifs > ModQV Versus Coverage by Motif

- Maps coverage against Modification QV for all genomic occurrences of a motif, for each reported motif.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Per-Base Kinetics:** CSV file containing per-base information.
- **Per-Base IPDs for IGV:** BigWig file containing encoded per-base IPD ratios.
- **Motif Annotations:** GFF file listing every modified nucleotide sequence motif in the genome.
- **Modified Base Motifs:** CSV file containing statistics for the methyltransferase recognition motifs detected.
- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.
- **Modified Bases:** GFF file listing every detected modified base in the genome.
- **Final Polished Assembly:** The polished assembly before oriC rotation is applied, in FASTA format.
- **Final Polished Assembly Index:** The BAI index file for the polished assembly before oriC rotation is applied.
- **Final Polished Assembly for NCBI:** The final polished assembly with applied oriC rotation and header adjustment for NCBI submission, in FASTA format.
- **Coverage Summary:** Coverage summary for regions (bins) spanning the reference sequence.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see ["Visualizing data using IGV" on page 118](#) for details.

- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.

- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.
- **Final Polished Assembly:** The polished assembly before oriC rotation is applied, in FASTA format.
- **Final Polished Assembly Index:** The BAI index file for the polished assembly before oriC rotation is applied.
- **Per-Base IPDs for IGV:** BigWig file containing encoded per-base IPD ratios.

Minor Variants Analysis

Use this application to identify and phase minor single nucleotide substitution variants in complex populations. This application is powered by the `juliet` algorithm:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Includes reference-based codon amino acid-calling (indel variants not called) in amplicons $\leq 4\text{kb}$, fully spanned by long reads.
- Includes extensive application reports for the HIV pol coding region, including drug resistance annotation from publicly-available databases.
- Includes reliable 1% minor variant detection with 6000 high-quality CCS reads with predicted accuracy of ≥ 0.99 per sample.
- The current version of this application provides additional reports for the HIV pol coding region, but it can be configured for **any** target organism or gene.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Set (Required)

- Specify a reference sequence to align the SMRT Cells reads to and to produce alignments.

Target Config (Required)

- Defines genes of interest within the reference and, optionally, drug resistance mutations for specific variants. Minor Variants Analysis contains one predefined target configuration for HIV HXB2. To specify this target configuration, enter `HIV_HXB2` into the **Target Config** field. To specify a **custom** target configuration for any organism or gene other than HIV HXB2: Enter **either** the path to the target configuration JSON file on the SMRT Link server, **or** the entire content of the JSON file.

Parameters

Advanced parameters	Default value	Description
Maximum Variant Frequency to Report (%) (Required)	100	Specify that only variants whose percentage of the population is less than this value be reported. Lowering this value helps to phase low-frequency variants when the highest frequency variant is different from the reference.
Minimum Variant Frequency to Report (%) (Required)	0.1	Specify that only variants whose percentage of the population is greater than this value be reported. Increasing this value helps to reduce PCR noise.

Advanced parameters	Default value	Description
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Phase Variants	ON	Specify whether to phase variants and cluster haplotypes.
Only Report Variants in Target Config	OFF	Specify whether to only report variants that confer drug resistance, as listed in the target configuration file.
Region of Interest	NONE	Specify genomic regions of interest; reads will be clipped to that region. If not specified, specifies all reads.
Target Config Override	NONE	If defined (and the main Target Config option is set to NONE), this string is interpreted as either a file system path to a JSON file, or the actual JSON content.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Minor Variants Analysis application generates the following reports:

Minor Variants > Summary

- **Barcode Name:** The pair of barcode indices for which the following metrics apply. If this was a single-sample analysis, this section of the report displays NA.
- **Median Coverage:** The median read coverage across all observed variant positions.
- **Number of Variants:** The number of variants found in the sample.
- **Number of Genes:** The number of genes observed in the sample.
- **Number of Affected Drugs:** The number of drugs to which resistance is conferred by variants in the sample.
- **Number of Haplotypes:** The number of haplotypes with different co-occurring variants found in the sample.
- **Maximum Frequency Haplotypes (%):** The maximum haplotype frequency reconstructed from the sample.

Minor Variants > Details

- **Barcode Name:** The pair of barcode indices for which the following metrics apply. If this was a single-sample analysis, this section of the report displays NA.
- **Position:** The amino acid position of the minor variant, with respect to the current gene.
- **Reference Codon:** The reference codon of the minor variant.
- **Variant Codon:** The mutated codon for the minor variant.
- **Variant Frequency (%):** The frequency of the minor variant, in percent.
- **Coverage:** The read coverage at the position of the codon.
- **ORF:** The name of the open reading frame/gene.
- **Affected Drugs:** Drugs to which resistance is conferred by the minor variant, according to a database specified in the configuration file.
- **Haplotypes:** The haplotypes associated with this variant.
- **Haplotype Frequencies (%):** The cumulative haplotype frequencies associated with the variant.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Variants Summary:** Data from the Minor Variants Details report, in CSV format.
- **Mapped Reads:** All input reads that were mapped to the reference by the application.
- **Detailed Reports:** Minor variants report information generated, as a ZIP-compressed HTML file. This includes the **full** report, in human-readable format, and contains four sections:

1. Input Data

Summarizes the data provided, the exact call for `juliet`, and `juliet` version for traceability purposes.

2. Target Config

Summarizes details of the provided target configuration for traceability. This includes the configuration version, reference name and length, and annotated genes. Each gene name (in bold) is followed by the reference start, end positions, and possibly known drug resistance mutations.

▼ Target config

Config Version: Predefined v1.1, PacBio internal

Reference Name: HIV_HXB2

Reference Length: 9719

Genes:

- **5'LTR** (1-634)
- **p17** (790-1186)
- **p24** (1186-1879)
- **p2** (1879-1921)
- **p7** (1921-2086)
- **p1** (2086-2134)
- **p6** (2134-2292)
- **Protease** (2253-2550)
 - ATV/r: V32I L33F M46I M46L I47V G48V G48M I50L I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88S L90M
 - DRV/r: V32I L33F I47V I47A I50V I54L I54M L76V V8F I84V
 - FPV/r: V32I L33F M46I M46L I47V I47A I50V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V L90M
 - IDV/r: V32I M46I M46L I47V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V N88S L90M
 - NFV: D30N L33F M46I M46L I47V G48V G48M I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88D N88S L90M
 - SQV/r: G48V G48M I54V I54T I54A I54L I54M V82A V82T I84V N88S L90M
 - TPV/r: V32I L33F M46I M46L I47V I47A I54V I54A I54M V82T V82L I84V

3. Variant Discovery

For each gene/open reading frame, there is one overview table.

Each row represents a variant position. Each variant position consists of the reference codon, reference amino acid, relative amino acid position in

the gene, mutated codon, percentage, mutated amino acid, coverage, and possible affected drugs.

Clicking the row displays counts of the multiple-sequence alignment counts of the -3 to +3 context positions.

▼ Variant Discovery

HIV HXB2			Reverse Transcriptase						
Codon	AA	Pos	Sample Variants				Affected Drugs*		
			AA	Codon	%	Coverage			
A T G	M	41	L	T T G	1	2793	ABC + DDI + TDF + D4T + ZDV		
A A A	K	65	R	A G A	1.1	2529	3TC + FTC + ABC + DDI + TDF + D4T		
			Pos	A	C	G	T	-	N
			-3	2947	0	0	0	0	51
			-2	2923	0	2	0	0	73
			-1	4	0	2952	0	0	42
			0	2606	0	0	0	339	53
			1	2905	0	29	0	0	64
			2	2938	0	0	0	0	60
			3	2938	0	0	0	0	60
			4	2942	0	0	0	0	56
			5	2751	0	0	0	0	247
T A T	Y	181	C	T G T	0.91	2946	NVP + EFV + ETR + RPV		
G G A	G	190	A	G C A	1	2947	NVP + EFV + ETR + RPV		
A C C	T	215	Y	T A C	0.93	2877	ABC + DDI + TDF + D4T + ZDV		

*HIVdb version 8.3 (last updated 2017-03-02)

► Legend

4. Drug Summaries

Summarizes the variants grouped by annotated drug mutations:

▼ Drug Summaries

Drug	Gene	Reference		Sample	
		AA	Pos	AA	%
3TC	Reverse Transcriptase	K	65	R	1
		M	41	L	0.99
		K	65	R	1
ABC	Reverse Transcriptase	T	215	Y	0.88

Phasing

The default mode is to call amino-acid/codon variants independently. Setting the **Phase Variants** parameter to **On**, variant calls from distinct haplotypes are clustered and visualized in the HTML output.

Protease										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
C G A	R	8	X	T G A	0.98	2931	MGI											

Reverse Transcriptase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A T G	M	41	L	T T G	0.99	2903	ABC + DDI + TDF + D4T + ZDV											
A A A	K	65	R	A G A	1	2577	3TC + FTC + ABC + DDI + TDF + D4T											
G G G	G	99	G	G G T	0.72	2907												
T T A	L	100	F	T T T	0.85	2819	MGI											
T A T	Y	181	C	T G T	0.95	2939	NVP + EFV + ETR + RPV											
G G A	G	190	A	G C A	1	2941	MGI + NVP + EFV + ETR + RPV											
A C C	T	215	Y	T A C	0.88	2940	ABC + DDI + TDF + D4T + ZDV											

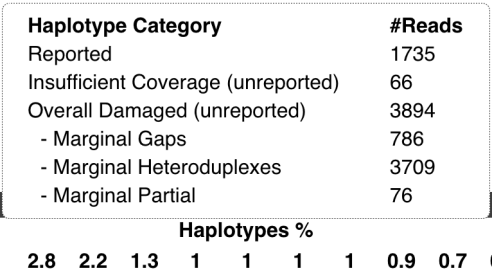
Integrase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A A A	K	188	K	A A G	0.92	2923	MGI											

- The row-wise variant calls are "transposed" onto per-column haplotypes. Each haplotype has an ID: [A-Z]{1}[a-z]{?}.
- For each variant, colored boxes in this row mark haplotypes that contain this variant.
- Colored boxes per haplotype/column indicate variants that co-occur. Wild type (no variant) is represented by plain dark gray. A color palette helps to distinguish between columns.
- The JSON variant positions has an additional `haplotype_hit` boolean array with the length equal to the number of haplotypes. Each entry indicates if that variant is present in the haplotype. A haplotype block under the root of the JSON file contains counts and read names. The order of those haplotypes matches the order of all `haplotype_hit` arrays.

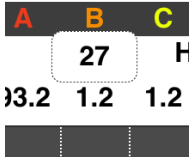
There are two types of tooltips in the haplotype section of the table.

The first tooltip is for the **Haplotypes %** and shows the number of reads that count towards (a) actually reported haplotypes, (b) haplotypes that have less than 10 reads and are not being reported, and (c) haplotypes that are not suitable for phasing. Those first three categories are mutually exclusive and their sum is the total number of reads going into `juliet`. For (c), the three different marginals provide insights into the sample

quality; as they are marginals, they are not exclusive and can overlap. The following image shows a sample with bad PCR conditions:



The second type of tooltip is for each haplotype percentage and shows the number of reads contributing to this haplotype:



Structural Variant Calling

Use this application to identify structural variants (Default: ≥ 20 bp) in a sample or set of samples relative to a reference. Variant types identified are insertions, deletions, duplications, copy number variants (CNVs), inversions, and translocations.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Set (Required)

- Specify a reference genome against which to align the reads and call variants.

Parameters

Advanced parameters	Default value	Description
Minimum Length of Structural Variant (bp) (Required)	20	The minimum length of structural variants, in base pairs.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Minimum % of Reads that Support Variant (any one sample) (Required)	10	Ignore calls supported by <P% of reads in every sample.
Minimum Reads that Support Variant (any one sample) (Required)	3	Ignore calls supported by <N reads in every sample.
Minimum Reads that Support Variant (total over all samples) (Required)	3	Ignore calls supported by <N reads total across samples.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Minimum Mapped Length (bp)	50	The minimum required mapped read length, in base pairs.
Minimum Gap-Compressed Identity (%)	70	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference.
Bio Sample Name of Aligned Dataset	NONE	Populates the Bio Sample Name (Read Group SM tag) in the aligned BAM file. If blank, uses the Bio Sample Name of the input file. Note: Avoid using spaces in Bio Sample Names as this may lead to third-party compatibility issues.

Advanced parameters	Default value	Description
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.0 for details.
Advanced pbsv Options	NONE	Additional pbsv command-line arguments. See SMRT® Tools reference guide v11.0 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

To launch a multi-sample analysis

1. Click + **Create New Job**.
2. Enter a **name** for the analysis.
3. Select all the Data Sets for all the input samples.
4. In the **Analysis of Multiple Data Sets** list, select **One Analysis for All Data Sets**.
5. Click **Next**.
6. Select **Structural Variant Calling** from the Analysis Application list.

Note: The Data Set field **Bio Sample Name** identifies which Data Sets belong to which biological samples.

- If **multiple** Data Sets with the same Bio Sample Name are selected and submitted, the Structural Variant Calling application **merges** those Data Sets as belonging to the same sample.
- If any input Data Sets do **not** have a Bio Sample Name specified, they are merged (if there are multiple such Data Sets) and their Bio Sample Name is set to `UnnamedSample` in the analysis results.

Reports and data files

The Structural Variant Calling application generates the following reports:

Report > Count by Sample (SV Type)

This table describes the type of called variants broken down by individual sample. For each sample, only variants for which the sample has a heterozygous ("0/1") or homozygous alternative ("1/1") genotype are considered.

- **Insertions (total bp):** The count and total length (in base pairs) of all called insertions in the sample.
- **Deletions (total bp):** The count and total length (in base pairs) of all called deletions in the sample.
- **Inversions (total bp):** The count and total length (in base pairs) of all called inversions in the sample.
- **Translocations:** The count of all called translocations in the sample.
- **Duplications (total bp):** The count and total length (in base pairs) of all called duplications in the sample.
- **Total Variants (total bp):** The count and total length (in base pairs) of all variants in the sample.

Report > Count by Sample (Genotype)

This table describes the genotype of called variants broken down by individual sample. For each sample, only variants for which the sample has a heterozygous ("0/1") or homozygous alternative ("1/1") genotype are considered.

- **Homozygous Variants:** The count of homozygous variants called in the sample.
- **Heterozygous Variants:** The count of heterozygous variants called in the sample.
- **Total Variants:** The count of all called variants in the sample.

Report > Count by Annotation

This table describes the called variants broken down by a set of repeat annotations. Each variant is counted once (regardless of sample genotypes) and assigned to exactly **one** annotation category. Only insertion and deletion variants are considered in this report.

- **Tandem repeat:** Variant sequence is a short pattern repeated directly next to itself.
- **ALU:** Variant sequence matches the ALU SINE repeat consensus.
- **L1:** Variant sequence matches the L1 LINE repeat consensus.
- **SVA:** Variant sequence matches the SVA LINE repeat consensus.
- **Unannotated:** Variant sequence does **not** match any of the above patterns.
- **Total:** The sum of variants from all annotations.

Report > Length Histogram

- Histogram of the distribution of variant lengths, in base pairs, broken down by individual. For each individual, separate distributions are provided for variants between 10-99 base pairs, 100-999 base pairs, and ≥ 1 kilobase pairs. Each variant is counted once, regardless of sample genotypes.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Aligned Reads (per sample):** Aligned reads, in BAM format, separated by individual.
- **Index of Aligned Reads (per sample):** BAM index files associated with the Aligned Reads BAM files.
- **Structural Variants:** All the structural variants, in VCF format.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see ["Visualizing data using IGV" on page 118](#) for details.

- **Aligned Reads (per sample):** Aligned reads, in BAM format, separated by individual.
- **Index of Aligned Reads (per sample):** BAM index files associated with the Aligned Reads BAM files.
- **Structural Variants:** All the structural variants, in VCF format. (See [here](#) for details.)

PacBio® data utilities

Following are data processing utilities provided with SMRT Analysis v11.0. These utilities are used as intermediate steps to producing biologically-meaningful results. Each utility is described later in this document, including all parameters, reports and output files generated by the utility.

Note: The following data utilities accept **only** HiFi reads as input.

5mC CpG Detection

- Analyze the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.
- See [“5mC CpG Detection” on page 91](#) for details.

Demultiplex Barcodes

- Separate reads by barcode.
- See [“Demultiplex Barcodes” on page 92](#) for details.

Export Reads

- Export HiFi reads that pass filtering criteria as FASTA, FASTQ and BAM files.
- For **barcoded** runs, you must **first** run the **Demultiplex Barcodes** application to create BAM files **before** using this application.
- See [“Export Reads” on page 98](#) for details.

Mark PCR Duplicates

- Remove duplicate reads from a HiFi reads Data Set created using an ultra-low DNA sequencing protocol.
- See [“Mark PCR Duplicates” on page 100](#) for details.

Trim Ultra-Low Adapters (was Trim gDNA Amplification Adapters)

- Trim PCR Adapters from a HiFi reads Data Set created using an ultra-low DNA sequencing library.
- See [“Trim Ultra-Low Adapters” on page 102](#) for details.

Note: The following data utility accept **only** Subreads as input.

Circular Consensus Sequencing (CCS)

- Identify consensus sequences for single molecules.
- See [“Circular Consensus Sequencing \(CCS\)” on page 104](#) for details.

5mC CpG Detection

Use this utility to analyze the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20. The utility also requires kinetics information.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Parameters

Advanced parameters	Default value	Description
Keep Kinetics in Output	OFF	If ON, specifies that the IPD and PulseWidth records are included in the output BAM file.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The 5mC CpG Detection utility generates the following reports:

5mC CpG Report > Methylation Probability

- **CpG Methylation in Reads:** The cumulative of percentage of CpG sites in the sample mapped against the predicted probability of methylation.
- **CpG Methylation in Reads (Histogram):** Histogram displaying the percentage of CpG sites in the sample versus the predicted probability of methylation.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **HiFi reads with 5mC Calls:** BAM file containing all the HiFi reads in the sample that include 5mC calls.
- **<Input Data Set>(5mC):** Output Data Set with the 5mC calls.

Demultiplex Barcodes

Use this utility to separate sequence reads by barcode. (See [“Working with barcoded data” on page 107](#) for more details.)

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Barcoded SMRTbell templates are SMRTbell templates with adapters flanked by barcode sequences, located on both ends of an insert.
- For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For **asymmetric** designs, **different** barcodes are attached to the sides of the insert sequence of interest.
- Barcode names and sequences, independent of orientation, **must** be unique.
- Most-likely barcode sequences per SMRTbell template are identified using a FASTA-format file of the known barcode sequences.

Given an input set of barcodes and a BAM Data Set, the Demultiplex Barcodes utility produces:

- A set of BAM files whose reads are annotated with the barcodes;
- A `ConsensusReadSet` file that contains the file paths of that collection of barcode-tagged BAM files and their related files.

Notes on Iso-Seq Multiplexed Data

There are two ways in which an Iso-Seq library can be multiplexed:

1. Barcoded adapter Iso-Seq libraries

- If using the SMRTbell Barcoded Adapter with the Iso-Seq Express protocol **on or after** April 21, 2022, demultiplex the Data Set **prior** to running the Iso-Seq application.
- To analyze the samples in a single Iso-Seq run, select **all** the demultiplexed Data Sets to combine and begin the Iso-Seq analysis.

2. Barcoded cDNA primer Iso-Seq libraries

- If following multiplexing guidelines using the Iso-Seq Express protocol **prior to** April 21, 2022 and you ordered synthesized oligos listed in the **Appendix 2 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section, demultiplex your Data Set using the Iso-Seq application. In other words, do **not** run the Demultiplexing Barcodes utility first.
- See [“Iso-Seq® Analysis” on page 67](#) for choices on Primer Sets to use.

Multiplexed method	Run Demultiplex Barcodes utility?
Not multiplexed	NO
Barcoded adapters	YES
Barcoded cDNA primer	NO

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Barcode Set (Required)

- Specify a barcode sequence file to separate the reads.

Same Barcodes on Both Ends of Sequence (Default = Yes)

- Specify **Yes** to retain all the reads with the **same** barcodes on both ends of the insert sequence, such as symmetric and tailed designs. (See [“Working with barcoded data” on page 107](#) for information on barcode designs.)
- Specify **No** to specify asymmetric designs where the barcodes are **different** on each end of the insert sequence.

Assign Bio Sample Names to Barcodes (Required)

SMRT Link automatically creates a CSV-format **Autofilled Barcoded Sample** File. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names 1 through N , corresponding to barcodes 1 through N , for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names to barcodes:

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the checkboxes to select multiple barcodes.)
- (Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid spaces in Bio Sample Names as they may lead to third-party compatibility issues.
- (Optional)** Click **Download as a file for later use**.
- Click **Save** to save the edited barcodes/Bio Sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a file:

1. Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
2. Select the **Barcoded Sample File** you just edited. You see **Success** on the line below, assuming the file is formatted correctly.

Demultiplexed Output Data Set Name (Required)

- Specify the name for the new demultiplexed Data Set that will display in SMRT Link. The utility creates a copy of the input Data Set, renames it to the name specified, and creates demultiplexed child Data Sets linked to it. The input data set remains separate and unmodified.

Parameters

Advanced parameters	Default value	Description
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Minimum Barcode Score	80	A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator of how well the chosen barcode pair matches. It ranges between 0 (no match) and 100 (a perfect match). Specifies that reads with barcode scores below this minimum value are not included in the analysis. This affects the output BAM file and the output demultiplexed Data Set XML file.
Advanced lima Options	NONE	Space-separated list of custom <code>lima</code> options. Not all supported command-line options can be used, and HPC settings cannot be modified. See the Demultiplex Barcodes section of the document SMRT® Tools reference guide v11.0 for information on <code>lima</code> .
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Demultiplex Barcodes utility generates the following reports:

Barcodes > Summary Metrics

- **Unique Barcodes:** The number of unique barcodes in the sequence data.
- **Barcoded Reads:** The number of correctly-barcoded reads in the sequence data.
- **Mean Reads:** The mean number of reads per barcode combination.
- **Max. Reads:** The maximum number of reads per barcode combination.
- **Min. Reads:** The minimum number of reads per barcode combination.
- **Mean Read Length:** The mean read length of reads per barcode combination.

- **Unbarcoded Reads:** The number of reads without barcodes in the sequence data.
- **Percent Bases in Barcoded Reads:** The percentage of bases in sequence data reads that contain barcodes.
- **Percent Barcoded Reads:** The percentage of reads in the sequence data that contain barcodes.

Barcodes > Barcode Data

- **Bio Sample Name:** The name of the biological sample associated with the barcode combination.
- **Barcode Name:** A string containing the pair of barcode indices for which the following metrics apply.
- **Polymerase Reads:** The number of polymerase reads associated with the barcode combination.
- **Bases:** The number of bases associated with the barcode combination.
- **Mean Read Length:** The mean read length of reads associated with the barcode combination.
- **Mean Barcode Quality:** The mean barcode quality associated with the barcode combination.

Barcodes > Inferred Barcodes

- **Barcode Name:** The barcode name.
- **Number of ZMWs:** The number of ZMWs out of the first 50,000 that are inferred to be assigned to the barcode combination.
- **Mean Barcode Score:** The mean barcode score associated with the reads inferred to be associated with the barcode combination.
- **Selected:** `Yes` if the number of ZMWs is at least 10, `No` otherwise.

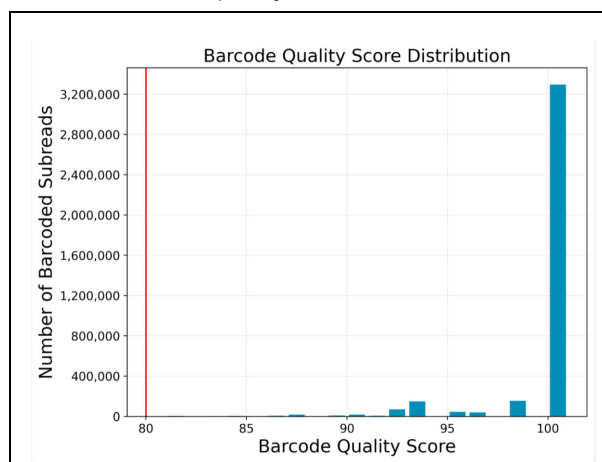
Barcodes > Barcoded Read Statistics

- **Number of Reads per Barcode:** Line graph displays the number of sorted reads per barcode.
 - **Good performance:** The Number of Reads per Barcode line (blue) should be mostly linear. Note that this depends on the choice of Y-axis scale. The mean Number of Reads per Barcode line (red) should be near the middle of the graph and should not be skewed by samples with too many or too few barcodes.
 - **Questionable performance:** A sharp discontinuity in the blue line, followed by no yield, with the red line way far from the center. Check the output file **Inferred Barcodes**, note the correct barcodes used, and consider reanalyzing the multiplexed samples with the correct Bio Sample names for the barcodes actually used. If you reanalyze the data, ensure that the **Barcode Name** file includes **only** the correct barcodes used.
- **Barcode Frequency Distribution:** Histogram distribution of read counts per barcode.
 - **Good performance:** A uniform distribution, which is most often a fairly tight symmetric normal distribution, with few barcodes in the tails.
 - **Questionable performance:** A large peak at zero. This can indicate use of incorrect barcodes. Check the output file **Inferred Barcodes**, note the correct barcodes used, and consider reanalyzing the multiplexed samples with the correct Bio Sample names for the barcodes actually used. If you reanalyze the data, ensure that the **Barcode Name** file includes **only** the correct barcodes used.
- **Mean Read Length Distribution:** Histogram distribution of the mean polymerase read length for all samples.
 - **Good performance:** The distribution should be normal with a relatively tight range.

- **Questionable performance:** A spread out distribution, with a mode towards the low end.

Barcodes > Barcode Quality Scores

- **Barcode Quality Score Distribution:** Histogram distribution of barcode quality scores. The scores range from 0-100, with 100 being a perfect match. Any significant modes or accumulation of scores <60 suggests issues with some of the barcode analyses. The red line is set at 80 – the minimum default barcode score.
- **Good performance:** HiFi demultiplexing runs should have >90% of reads with barcode quality score ≥ 95 .



- **Questionable performance:** A bimodal distribution with a large second peak usually indicates that some barcodes that were sequenced were **not** included in the barcode scoring set.

Barcodes > Barcoded Read Binned Histograms

- **Read Length Distribution By Barcode:** Histogram distribution of the polymerase read length by barcode. Each column of rectangles is similar to a read length histogram rotated vertically, seen from the top. Each sample should have similar polymerase read length distribution. Non-smooth changes in the pattern looking from left to right might indicate suboptimal performance.
- **Barcode Quality Distribution By Barcode:** Histogram distribution of the per-barcode version of the **Read Length Distribution by Barcode** histogram. The histogram should contain a single cluster of hot spots in each column. All barcodes should also have similar profiles; significant differences in the pattern moving from left to right might indicate suboptimal performance.
 - **Good performance:** All columns show a single cluster of hot spots.
 - **Questionable performance:** A bimodal distribution would indicate missing barcodes in the scoring set.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **All Barcodes (FASTA):** All barcoded reads, in FASTA format.

- **Barcode Files:** Barcoded subread Data Sets; one file per barcode.
- **Barcoding Summary CSV:** Data displayed in the reports, in CSV format. This includes Bio Sample Name.
- **Barcode Summary:** Text file listing how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- **Inferred Barcodes:** Inferred barcodes used in the analysis. The barcoding algorithm looks at the first 35,000 ZMWs, then selects barcodes with ≥ 10 counts and mean scores ≥ 45 .
- **Unbarcoded Reads:** BAM file containing reads not associated with a barcode.
- **demultiplex.<barcode>.hifi.reads.fastq.gz:** Gzipped HiFi reads in FASTQ format, one file per barcode.

Export Reads Use this utility to export HiFi reads that pass filtering criteria as FASTA, FASTQ and BAM files.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- For **barcoded** runs, you must **first** run the **Demultiplex Barcodes** utility to create BAM files **before** using this utility.
- This utility does **not** generate any reports.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Output FASTA File (Default = ON)

- Outputs a single FASTA/FASTQ file containing all the reads that passed the filtering criteria.

Output BAM File (Default = OFF)

- Outputs a single BAM file containing all the reads that passed the filtering criteria.

Min. CCS Predicted Accuracy (Phred Scale) Default = 20

- Phred-scale integer QV cutoff for filtering HiFi reads. The default for **all** applications is 20 (QV 20), or 99% predicted accuracy.

Parameters

Advanced parameters	Default value	Description
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **hifi_reads.fasta.gz:** Sequence data that passed filtering criteria, converted to Gzipped FASTA format.

- **hifi_reads.fastq.gz**: Sequence data that passed filtering criteria, converted to Gzipped FASTQ format.
- **<Reads>.bam**: Sequence data that passed filtering criteria.

Mark PCR Duplicates

Use this utility to remove duplicate reads from a **HiFi reads** Data Set created using an ultra-low DNA sequencing protocol.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Note: If starting with a very low-input DNA sample using the **SMRTbell gDNA sample amplification kit**, you **must** run this utility (preceded by the **Trim Ultra-Low Adapters** utility) on the resulting Data Set **prior** to running any secondary analysis application.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Parameters

Advanced parameters	Default value	Description
Identify Duplicates Across Sequencing Libraries	ON	Duplicate reads are identified per sequencing library. The library is specified in the BAM read group LB tag, which is set using the Well Sample Name field in Run Design. By convention, different LB tags correspond to different library preparations. Use this option when the LB tag does not follow this convention to treat all reads as from the same sequencing library.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Mark PCR Duplicates utility generates the following reports:

PCR Duplicates > Duplicate Rate (table)

- Library:** The name of the library containing duplicate molecules.
- Unique Molecules:** The number of unique molecules in the library.
- Unique Molecules (%):** The percentage of unique molecules in the library.
- Duplicate Reads:** The number of duplicate reads in the library.
- Duplicate Reads (%):** The percentage of duplicate reads in the library.

PCR Duplicates > Duplicate Rate (chart)

- Duplicate Rate:** Displays the percentage of duplicate reads per library.
- Duplicate Reads per Molecule:** Displays the percentage of duplicated molecules per library; broken down by the number of reads per duplicated molecule.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **PCR Duplicates:** BAM file containing duplicate reads with PCR adapters.
- **<Data Set> (deduplicated):** Output Data Set, with duplicate reads with PCR adapters removed.

Trim Ultra-Low Adapters

Use this utility to trim PCR Adapters from a HiFi reads Data Set created using an ultra-low DNA sequencing library.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Note: If starting with a very low-input DNA sample using the **SMRTbell gDNA sample amplification kit**, you **must** run this utility (followed by the **Mark PCR Duplicates** utility) on the resulting Data Set **prior** to running any secondary analysis application.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

PCR Adapters (Required)

- Specify the file of PCR adapters used during library preparation of an ultra-low DNA sequencing library to be trimmed from the sequenced data.

Parameters

Advanced parameters	Default value	Description
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Trim Ultra-Low Adapters utility generates the following reports:

PCR Adapters > Summary Metrics

- **Unique PCR Adapters:** The number of unique PCR adapters in the sequence data.
- **Reads with PCR Adapters:** The number of reads in the sequence data that contain PCR adapters.
- **Mean Reads Per Adapter:** The mean number of reads per PCR adapter in the sequence data.
- **Max. Reads Per Adapter:** The maximum number of reads per PCR adapter in the sequence data.
- **Min. Reads Per Adapter:** The minimum number of reads per PCR adapter in the sequence data.
- **Mean Read Length:** The mean read length of reads per PCR adapter in the sequence data.

- **Reads Without PCR Adapters:** The number of reads without PCR adapters in the sequence data.
- **Percent Bases in Reads with Adapters:** The percentage of bases in reads in the sequence data that contain PCR adapters.
- **Percent Reads with Adapters:** The percentage of reads in the sequence data that contain PCR adapters.

PCR Adapters > PCR Adapter Data

- **Bio Sample Name:** The name of the biological sample associated with the PCR adapters.
- **PCR Adapter Name:** A string containing the pair of PCR adapter indices for which the following metrics apply.
- **Polymerase Reads:** The number of polymerase reads associated with the PCR adapter.
- **Bases:** The number of bases associated with the PCR adapter.
- **Mean Read Length:** The mean read length of reads associated with the PCR adapter.
- **Mean PCR Adapter Quality:** The mean PCR adapter quality associated with the PCR adapter.

PCR Adapters > PCR Adapter Read Statistics

- **Number of Reads Per PCR Adapter:** Histogram distribution of the mean number of reads per PCR adapter.
- **PCR Adapter Frequency Distribution:** Histogram distribution of reads with PCR adapter mapped to the number of barcoded samples.
- **Mean Read Length Distribution:** Maps the mean read length against the number of barcoded samples.

PCR Adapters > PCR Adapter Quality Scores

- Histogram distribution of PCR adapter quality scores. The scores range from 0-100, with 100 being a perfect match.

PCR Adapters > PCR Adapter Read Binned Histograms

- **Read Length Distribution By PCR Adapter:** Histogram distribution of the read length by PCR adapter. Each column of rectangles is similar to a read length histogram rotated vertically, seen from the top.
- **PCR Adapter Quality Distribution By Barcode:** Histogram distribution of the per-barcode version of the **Read Length Distribution by PCR Adapter** histogram.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Reads Missing Adapters:** Reads Missing Adapters: BAM file containing the reads with missing PCR adapters from the input Data Set.
- **PCR Adapter Data CSV:** Includes the data displayed in the PCR Adapter Data table.
- **<Data Set> (trimmed):** Output Data Set, with the PCR adapters removed.

Circular Consensus Sequencing (CCS)

Use this utility to identify consensus sequences for single molecules.

- The utility accepts **Subreads** (BAM format) as input.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Detect 5mC Sites (Default = OFF)

- If set to **ON**, kinetics analysis to identify 5mC CpG sites will be performed.

Parameters

Advanced parameters	Default value	Description
Minimum CCS Read Length	10	The minimum length for the median size of insert reads to generate a consensus sequence. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates.
Maximum CCS Read Length	50,000	The maximum length for the median size of insert reads to generate a consensus sequence. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates.
Generate a Consensus for Each Strand	OFF	Generate a consensus for each strand. Warning: This is an experimental option for the CCS algorithm, and may not be compatible with all downstream applications. We recommend using command-line analysis for this feature.
Process All Reads	OFF	Specifies behavior identical to on-instrument CCS reads generation, overriding all other cutoffs. This setting writes a CCS read for every ZMW in the input Data Set. Set to OFF to specify more restrictive settings.
Include Kinetics Information with CCS Analysis output	OFF	If ON , include kinetics per-base data required for methylation DNA analysis. Note: This results in a BAM file that is 3-4 times larger. This option applies only when Process All Reads is set to ON .
Advanced CCS Options	NONE	Space-separated list of additional command-line options to CCS analysis. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.0 for details.
Minimum Predicted Accuracy (Deprecated)	0.99	The minimum predicted accuracy of a read, ranging from 0 to 1. (0.99 indicates that only reads expected to be 99% accurate are emitted.) Note: This setting is ignored if the Process All Reads advanced parameter is set to ON .
Minimum Number of Passes (Deprecated)	3	The minimum number of full passes for a ZMW to be used. Full passes must have an adapter hit before and after the insert sequence and so do not include any partial passes at the start and end of the sequencing reaction. Note: This setting is ignored if the Process All Reads advanced parameter is set to ON .
Detect And Split Heteroduplex Read	OFF	Specifies that any detected heteroduplexes are separated into separate reads.

Advanced parameters	Default value	Description
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Circular Consensus Sequencing (CCS) utility generates the following reports:

CCS Analysis Report > Summary Metrics

Note: CCS reads with quality value equal to or greater than 20 are called **HiFi reads**.

- **HiFi Reads:** The total number of CCS reads whose quality value is equal to or greater than 20.
- **HiFi Yield (bp):** The total yield (in base pairs) of the CCS reads whose quality value is equal to or greater than 20.
- **HiFi Read Length (mean, bp):** The mean read length of the CCS reads whose quality value is equal to or greater than 20.
- **HiFi Read Quality (median):** The median number of CCS reads whose quality value is equal to or greater than 20.
- **HiFi Number of Passes (mean):** The mean number of passes used to generate CCS reads whose quality value is equal to or greater than 20.

CCS Analysis Report > HiFi Read Length Summary

- **Read Length (bp):** The HiFi read length, ranging from ≥ 0 to $\geq 40,000$ base pairs.
- **Reads:** The number of HiFi reads with the specified read length.
- **Reads (%):** The percentage of HiFi reads with the specified read length.
- **Yield (bp):** The number of base pairs in the HiFi reads with the specified read length.
- **Yield (%):** The percentage of base pairs in the HiFi reads with the specified read length.

CCS Analysis Report > HiFi Read Quality Summary

- **Read Quality (Phred):** Phred-scale quality values, ranging from QV ≥ 20 to QV ≥ 50 .
- **Reads:** The number of HiFi reads with the specified read quality.
- **Reads (%):** The percentage of HiFi reads with the specified read quality.
- **Yield (bp):** The number of base pairs in the HiFi reads with the specified read quality.
- **Yield (%):** The percentage of base pairs in the HiFi reads with the specified read quality.

CCS Analysis Report > Read Length Distribution

- **HiFi Read Length Distribution:** Histogram distribution of HiFi reads by read length.
- **Yield by HiFi Read Length:** Histogram distribution of the cumulative yields of CCS reads by read length.
- **Read Length Distribution:** Histogram distribution of all reads by read length.

CCS Analysis Report > Number of Passes

- Histogram of the number of complete subreads in CCS reads, broken down by number of reads.

CCS Analysis Report > Read Quality Distribution

- Histogram distribution of the CCS reads by the Phred-scale read quality.

CCS Analysis Report > Predicted Accuracy vs. Read Length

- Heat map of CCS read lengths and predicted accuracies.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **CCS Analysis Per-Read Details:** Summary of CCS analysis performance and yield.
- **hifi_reads.fastq.gz:** Gzipped HiFi reads in FASTQ format.
- **hifi_reads.fasta.gz:** Gzipped HiFi reads in FASTA format.
- **hifi_reads.bam:** HiFi reads in BAM format.
- **All Reads (BAM):** BAM file containing one CCS read per ZMW, including the following types of reads:
 - HiFi reads (Q20 or higher)
 - Lower-quality but still polished consensus reads (Q1-Q20)
 - Unpolished consensus reads (RQ=-1)
 - 0- or 1-pass subreads unaltered (RQ=-1)

Working with barcoded data

This section describes how to use SMRT Link to work with barcoded data. Demultiplex Barcodes analysis is powered by the `lima` SMRT Analysis tool.

The canned data provided with SMRT Link v11.0 includes 7 barcode sets:

- gDNA Amplification Adapter
- Iso-Seq 12 Barcoded cDNA Primers
- Iso-Seq cDNA Primers
- Barcoded Overhang Adapter Kits 8A and 8B
- Sequel 384 barcodes v1
- Barcoded M13 Primer Plate
- SMRTbell Barcoded Adapter Plate 3.0

SMRT Link v11.0 supports sample traceability through the various modules in the application by using the **Bio Sample Name**.

Run Design in SMRT Link v11.0 contains a required **Bio Sample Name** field for both single and multiplexed samples.

- For multiplexed experiments, SMRT Link provides default names for **one** Bio Sample Name per barcode, which can be edited as needed in the **Barcoded Sample Names** file.
- For multiplexed Iso-Seq Analysis **only**, Bio Sample Names are **not** required.

Well Sample Name and **Bio Sample Name** entered in Sequel II systems, and in Run Designs for multiplexed runs:

- Display as column values in the Data Management and SMRT Analysis modules.
- Display as Data Set attributes in the Data Set details page in Data Management.
- Populate the `LB` and `SM` tags in read group headers of BAM files containing basecalled data.

Example Well Sample Names and Bio Sample Names

- **Non-barcoded Well Sample Name:** HG002_2019_11_2_10K
- **Non-barcoded Bio Sample Name:** HG002
- **Barcoded Well Sample Name:** My Multiplexed Set of Bugs
- **Barcoded Bio Sample Name:**
Unknown Microbe 1,...,Unknown Microbe N

Step 1: Specify the barcode setup and sample names in a Run Design

Note: If you specified the barcode setup in Run Design, the demultiplexing is performed **automatically** after the data is transferred to the SMRT link server. You can also specify the barcode setup **manually** by selecting **SMRT Analysis > Create New Job** and then selecting the Demultiplex Barcodes data utility.

1. In SMRT Link, create a new Run Design as described in “[Creating a new Run Design](#)” on page 16. **Before** you finish the new Run Design, perform the following steps.

Barcoded Sample Options

Sample Is Barcoded ☒ YES ☐ NO

Barcode Set Required Sequel_16_barcode_v3

Same Barcodes on Both Ends of Sequence ☒ YES ☐ NO

Assign Bio Sample Names to Barcodes Required Interactively From a File

Autofilled Barcoded Sample File [Download File](#)

Barcoded Sample Name File Required Choose file Browse

2. Click **Barcoded Sample Options** and then click **Yes** for **Sample is Barcoded**. Additional fields related to barcoding display.
3. Specify a **Barcode Set** using the dropdown list.
Note: You can specify up to 10,000 samples. Specifying **more** than 10,000 samples may cause a delay of several minutes in analysis submission.
4. Specify if the **same** barcodes are used on both ends of the sequences.
 - Selecting **Yes** specifies symmetric and tailed designs where **all** the reads have the same barcodes on both ends of the insert sequence. Barcode analysis of such experiments retains **only** data with the same barcode identified on both ends.
 - Selecting **No** specifies asymmetric designs where the barcodes are **different** on each end of the insert. Barcode analysis of such experiments retains any barcode pair combination identified in the Data Set.
5. SMRT Link automatically creates a CSV-format **Autofilled Barcode Name** file. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names 1 through N, corresponding to barcodes 1 through N, for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names to barcodes. (**Note:** Bio Sample Names are hardcoded and can be traced through secondary analysis using SMRT Analysis.)

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the check boxes to select multiple barcodes.)

- **(Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid using spaces in Bio Sample Names as they may lead to third-party compatibility issues.
- **(Optional)** Click **Download as a file for later use**.
- Click **Save** to save the edited barcodes/Bio Sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a File:

- Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
 - Select the **Barcoded Sample** file you just edited. You see **Success** on the line below, assuming the file is formatted correctly.
6. Specify **if** and **where** to automatically generate **HiFi reads** (reads generated with CCS analysis whose quality value is equal to or greater than 20):
- **On Instrument** (available **only** for the Sequel IIe system): HiFi reads are automatically generated on the instrument, **before** transfer to the compute cluster where SMRT Link is installed.
 - **In SMRT Link**: HiFi reads are automatically generated **after** transfer to the compute cluster where SMRT Link is installed.
 - **Do Not Generate**: HiFi reads are **not** generated for this run. Only subread data are transferred to the local compute cluster where SMRT Link is installed.
7. Click **Save**.

Step 2: Perform the sequencing run

Load the samples and perform the sequencing run, using the Run Design you created in Step 1. The demultiplexing analysis is performed automatically on the SMRT Link server once the data is transferred from the Sequel II systems. This creates an analysis of type `Demultiplex Barcodes (Auto)` in the SMRT Analysis module. You can click to select this analysis and review the reports and data created. If everything looks fine, you can continue to **Step 4** and use the demultiplexed Data Set(s) created by the run as input to further analysis.

Note: By default, `Demultiplex Barcodes (Auto)` creates **one** Data Set per autodetected barcode within the selected barcode set. It also applies a Data Set filter of a minimum barcode score greater than 26 for optimal results in secondary analysis. If used, the analysis parameter **Filters to add to the Data Set** overrides other barcode filtering, even if the barcode score set with it is lower than 26.

**Step 3:
(Optional) Run
the Demultiplex
Barcodes data
utility**

If instead you did **not** specify the barcode setup in the Run Design, or if you need to change any of the parameters used in the **Demultiplex Barcodes** analysis automatically launched from Run Design, run the **Demultiplex Barcodes** data utility. This separates reads by barcode and creates a new demultiplexed Data Set that you can then use as input to other secondary analysis applications.

1. Click **+ Create New Job**.
2. Enter a **name** for the job.
3. Select **Data Utility** as the workflow type.
4. Select **HiFi reads** as the data type to use. The Data Sets table displays the appropriate Data Sets available for the job.
5. In the Data Sets table, select one or more Data Sets to be analyzed together.
6. Click **Next**.
7. Select **Demultiplex Barcodes** from the Applications list.
8. Specify a **Barcode Set** (barcode sequence file.)
Note: You can specify up to 10,000 samples. Specifying **more** than 10,000 samples may cause a delay of several minutes in analysis submission.
9. Specify if the **same** barcodes are used on both ends of the sequences.
 - Selecting **Yes** specifies symmetric and tailed designs where **all** the reads have the same barcodes on both ends of the insert sequence. Barcode analysis of such experiments retains **only** data with the same barcode identified on both ends.
 - Selecting **No** specifies asymmetric designs where the barcodes are **different** on each end of the insert. Barcode analysis of such data retains any barcode pair combination identified in the Data Set.
10. SMRT Link automatically creates a CSV-format **Autofilled Barcoded Sample** file. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names **1** through **N**, corresponding to barcodes **1** through **N**, for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names to barcodes:

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the check boxes to select multiple barcodes.)
- **(Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid using spaces in Bio Sample Names as they may lead to third-party compatibility issues.
- **(Optional)** Click **Download as a file for later use**.
- Click **Submit** to save the edited barcodes/bio sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a File:

- Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
 - Select the **Barcoded Sample** file you just edited. You see **Success** on the line below, assuming the file is formatted correctly.
11. Specify the **name** for the new demultiplexed Data Set that will display in SMRT Link. The application creates a copy of the input Data Set, renames it to the name specified, and creates demultiplexed child Data Sets linked to it. The input Data Set remains separate and unmodified.
 12. (**Optional**) Specify any advanced parameters.
 13. Click **Start**. After the analysis is finished, a new demultiplexed Data Set is available.

Note: For information about the reports generated by the Demultiplex Barcodes data utility, see [“Demultiplex Barcodes” on page 90](#).

Step 4: Run applications using the demultiplexed data as input

All secondary analysis applications except **Demultiplex Barcodes** can use demultiplexed Data Sets as input.

Note: For **Iso-Seq** analysis with barcoded samples, use the Iso-Seq application **instead** of the Demultiplex Barcodes data utility, as the Iso-Seq application **already** includes the demultiplexing step as part of the pipeline. When performing multiplexed Iso-Seq analysis, ensure that the Run Design **Sample Is Barcoded** option is set to **No** (the default setting). Then, in SMRT Analysis, go straight to the Iso-Seq application and, in the parameters section, select a Primer Set containing multiple primers, such as `IsoSeq_Primers_12_Barcodes_v1`.

1. Select the secondary analysis application/data utility to use.
2. Click the number in the **Demultiplexed Subsets** column, then select the demultiplexed Data Set to use as input:

Data Sets

<input type="checkbox"/>	Name	Demultiplexed Subsets
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles	3

Members of Re-barcode Alice/Bob/Charles

<input type="checkbox"/>	Name	Well Sample Name
<input checked="" type="checkbox"/>	Re-barcode Alice/Bob/Charles (Charles)	T0213_384-plex_barcode_A01
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles (Bob)	T0213_384-plex_barcode_A01
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles (Alice)	T0213_384-plex_barcode_A01

– You can select the **entire** Data Set as input, or one or more specific outputs from selected barcodes, to a maximum of 16 sub-Data Sets, 12 for Iso-Seq.

3. Additional **Analysis Type** options become available. You can select from the following options:

Workflow Type

☒ ANALYSIS ☐ AUTO ANALYSIS ☐ DATA UTILITY

Analysis of Multiple Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis for All Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis per Data Set - Custom Parameters

- **One Analysis for All Data Sets:** Runs one job using all the selected barcode Data Sets as input, for a maximum of 30 Data Sets.
- **One Analysis per Data Set - Identical Parameters:** Runs **one** separate job for **each** of the selected barcode Data Sets, using the **same** parameters, for a maximum of 10,000 Data Sets. Optionally click **Advanced Parameters** and modify parameters.
- **One Analysis per Data Set - Custom Parameters:** Runs **one** separate job for **each** of the selected barcode Data Sets, using **different** parameters for each Data Set, for a maximum of 16 Data Sets. Click **Advanced Parameters** and modify parameters. Then click **Start and Create Next**. You can then specify parameters for each of the included barcode Data Sets.
- **Note:** The number of Data Sets listed is based on testing using PacBio's suggested compute configuration, listed in **SMRT Link software installation guide (v11.0)**.

4. Click **Start** to submit the job.

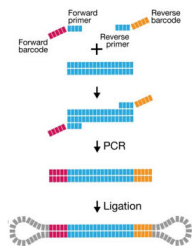
Demultiplex Barcodes details

The **Demultiplex Barcodes** data utility identifies barcode sequences in PacBio single-molecule sequencing data.

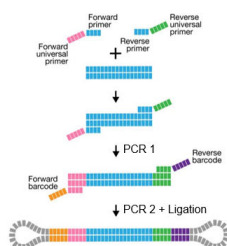
Demultiplex Barcodes can demultiplex samples that have a unique per-sample barcode pair and were pooled and sequenced on the same SMRT Cell. There are four different methods for barcoding samples with PacBio technology:

- 1. Barcoded target-specific primers
- 2. Barcoded universal primers
- 3. Barcoded overhang adapters
- 4. Barcoded linear adapters (target capture)

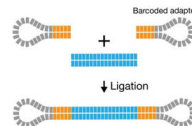
1. Barcoded Target-Specific Primers



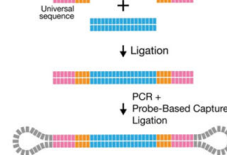
2. Barcoded Universal Primer



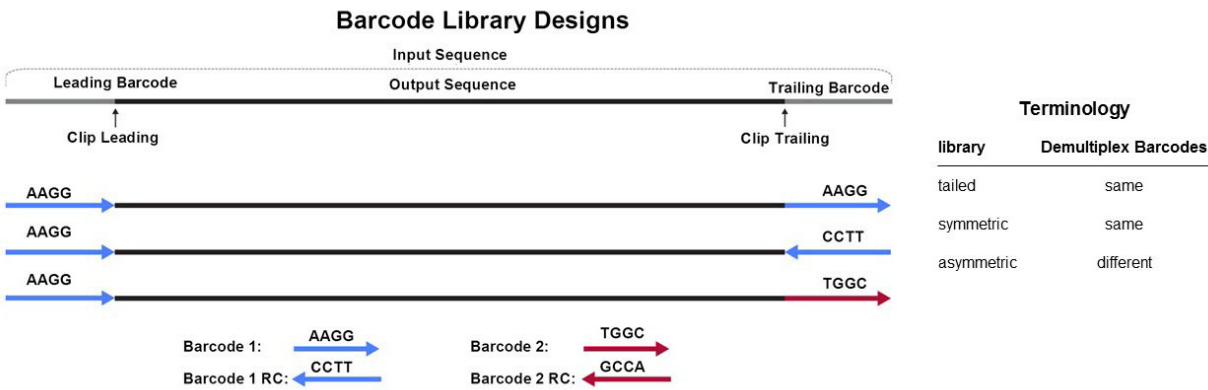
3. Barcoded Overhang Adapters



4. Barcoded Linear Adapter (Target Capture)



In addition, there are three different barcode library designs.



The **Demultiplex Barcodes** application in SMRT Link supports demultiplexing of subreads.

Demultiplexing of CCS reads is possible using the command line. (See **SMRT® Tools reference guide (v11.0)** for details.)

Symmetric mode

For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For barcode identification, one read with a single barcode region is sufficient. Symmetric barcoding is used for samples constructed using Barcoded overhang adapters, Barcoded universal primer and target enrichment (linear). This is also the default scoring mode in SMRT Link v10.2 and later.

Asymmetric mode

Barcode sequences are **different** on the ends of the SMRTbell template. Asymmetric mode is used with the M13 barcoding procedure. (See the document **Procedure & checklist - Preparing SMRTbell libraries using PacBio barcoded M13 primers for multiplex SMRT sequencing** for details.) PacBio using this mode **only** for small inserts (up to 5 kb) where both ends of the insert are expected to be sequenced. **Both** barcodes must be detected.

Note: For **both** Symmetric and Asymmetric modes, the limit for unique individual barcode sequences is 768, and the limit for the number of different barcode pairs is 10,000.

When running the **Demultiplex Barcodes** data utility in SMRT Link, set the **Same Barcodes on Both Ends of the Sequence** option to **Off**.

Mixed mode

Libraries with combined symmetric and asymmetric barcoding are **not** supported.

Automated analysis

Auto Analysis and **Pre Analysis** allow a specific analysis to be **automatically** run after a sequencing run has finished and the data is transferred to the SMRT Link server. The analysis can include demultiplexed output.

- Auto Analysis can be set up in Run Design or SMRT Analysis **after** the Run Design is saved and **before** the run is loaded on the instrument.
- Auto Analysis can be run on HiFi reads, and includes **all** analysis applications available.
- Auto Analysis works with **all** Sequel II systems.

Pre Analysis is the process of CCS analysis and/or demultiplexing of Sequel basecalled data. Pre Analysis occurs **before** Auto Analysis, and is defined when you create a Run Design and specify one or more of the following:

- Read Type = **HiFi reads** and Generate HiFi reads = **On Instrument** or **In SMRT Link**.
- Read Type = **HiFi reads** and Sample is Barcoded = **Yes**.

Note: Pre Analysis works with **all** Sequel II systems.

Creating an Auto Analysis job from SMRT Analysis

This procedure includes only the basic steps - for more detailed information on creating jobs, see [“Creating and starting a job” on page 44](#).

1. Access SMRT Link using the Chrome web browser.
2. Select **SMRT Analysis**.
3. Click **+ Create New Job**.
4. Enter a **name** for the job.
5. Specify **Auto Analysis** as the workflow type. The table displays available runs. (**Note:** Runs display here **only** if they are in the **Created** state - not if they are already running or have completed.)
6. Click a **Collections** link associated with a specific run in the table.
7. Select **one** sample, or drill down further and select from the barcoded samples of a single collection by clicking the **Barcoded Samples** link. You can select **multiple** barcoded samples at this level.
Note: You **cannot** select a mix of samples and barcoded samples, or select barcoded samples from **multiple** samples. In addition, samples or barcoded samples will **not** be selectable if they have a Job Id.
8. Click **Next**.
9. Select a secondary analysis application from the dropdown list.
Note: **Only** analysis applications, **not** data utilities, are available for use with Auto Analysis.
10. (**Optional**) Click **Advanced Parameters** and specify the values of the parameters to change. Click **OK** when finished. (Different applications have different advanced parameters.)

- To see information about parameters for **all** secondary analysis applications provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#).

11. Click **Start** to submit the Auto Analysis job.

Creating Auto Analysis from a Run Design

1. Create a new Run Design (See [“Creating a new Run Design” on page 16](#) for details) and save it. The **Auto Analysis** button is enabled only **after** you save the Run Design.
2. Click **Auto Analysis**. This takes you into SMRT Analysis, where you create the new job that will be associated with the collection.
3. Name the new job.
4. Click the numbered **Collections** link (Column 2 of the Runs table) associated with the run that you defined in Step 1. (**Note:** Runs display here **only** if they are in the **Created** state - not if they are already running or have completed.)
5. Select a collection for analysis.
6. Click **Next**.
7. Select a secondary analysis application to use for the analysis.
8. (**Optional**) Click **Advanced Parameters** and specify the values of the parameters to change. Click **OK** when finished. To see information about parameters for **all** secondary analysis applications provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#).
9. Click **Create**.

HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design

The **HiFiViral SARS-CoV-2** Application includes a streamlined version of Auto Analysis as part of creating a Run Design.

1. In Run Design, click **Create New Design**.
2. From the Application list, select **HiFiViral SARS-CoV-2**. Preloaded default values display in green.
3. Enter the **Well Sample Name**.
4. By default, **Auto Analysis** is set to **Yes** for **Automatic Launch of SARS-CoV-2 Analysis**.
5. Enter the **Analysis Name**.
6. By default, **Yes** is selected for **Sample Is Barcoded** and **No** for **Same Barcode on Both End of Sequence**.
7. By default, the barcode set **HiFiViral SARS-CoV-2 M13barcodes** is selected.
8. For **Assign Bio Sample Names to Barcodes**, select **From a File** and then click **Download File**.
9. Open the downloaded file (**assayPlateQC_template_4by96.csv**) in a text editor. See [“HiFiViral SARS-CoV-2 Analysis” on page 54](#) for details on modifying this file for your samples.

Getting information about analyses created by Auto Analysis

There are several ways to obtain information on the state of an analysis created using the Auto Analysis feature.

From SMRT Analysis:

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. To filter the jobs, click the funnel in the **State** column header, then click **Created**. This displays **only** jobs in the **Created** state.
3. Click the job of interest.
4. Click the **From Multi-Job** link.
5. Click **Analysis Overview > Status of Individual Analyses**. This displays information about the analysis, including the application used.

From Run Design:

1. On the home page, select **Run Design**.
2. Click the Run Design of interest.
3. Click the **From Multi-Job** link.
4. Scroll all the way to the right in the table. This displays information about the samples included in the run.
5. Click the **Auto Analysis ID** link for a sample. This displays information about the analysis, including the application used.

Getting information about Pre Analysis from SMRT Analysis

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. To filter the jobs, click the funnel in the **State** column header, then click **Created**. This displays **only** jobs in the **Created** state.
3. Click the job of interest.
4. Click the **Pre Analysis** link.
5. Click **Analysis Overview > Status of Individual Analyses**. This displays information about the Pre Analysis, including the application used.

Getting information about Pre Analysis from Run Design

1. On the home page, select **Run Design**.
2. Click the Run Design of interest.
3. On the left side (above the consumables list), click the **Pre Analysis ID** link. This displays information about the Pre Analysis, including the application used.

Visualizing data using IGV

Once an analysis has successfully completed, visualize the results using the **Integrative Genomics Viewer (IGV)**.

- See [here](#) for further installation instruction and usage details.
- See [here](#) for PacBio-specific settings and visualizations.

You can visualize data generated by the following secondary analysis applications:

- Iso-Seq Analysis
- HiFi Mapping
- Microbial Genome Analysis
- Structural Variant Calling

IGV requires the following files for visualization:

- One consolidated alignment BAM file
- BAM index file
- Genome reference file

If an analysis generates **multiple** alignment BAM files, those files must **first** be combined into **one** consolidated alignment BAM file for visualization with IGV.

SMRT Link **defaults** to combining chunked alignment BAM files if the combined file sizes are **10 GB or less**.

- When creating an analysis, you can specify that SMRT Link combines alignment BAM files for IGV visualization by setting the **Consolidate Mapped BAMs for IGV** option to **ON**.

Note: This setting **doubles** the amount of storage used by the BAM files, which can be considerable. Make sure to have enough disk space available. This setting may also result in longer run times.

To visualize data using IGV

1. Create and run the analysis.
2. After the analysis has finished successfully, go to the **Data > IGV Visualization Files** section of the analysis report page.
3. Open IGV and select the reference genome used for the analysis. (See [here](#) for instructions on how to load a genome.)
4. Copy a BAM file link from the **Data > IGV Visualization Files** section of the analysis report page.

Note: If you are performing *de novo* assembly, you **must** use links to the **draft** assembly BAM files, which are clearly labeled.

IGV Visualization Files

File	Path	Size	Type
Draft Assembly	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-collect_configs/execution/collected_ctg.fasta	20 KB	Fasta
Mapped BAM	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-auto_consolidate_alignments/execution/mapped.bam	470 KB	bam
Mapped BAM Index	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-auto_consolidate_alignments/execution/mapped.bam.bai	336 bytes	bam_bai
Draft Assembly Index	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-make_faidx/execution/job-fad5cbaf8543cbe1e94fb4cdcbef5d56/collected_ctg.fasta.fai	107 bytes	SamIndex

Click to copy the file path to the Clipboard.

5. In IGV, choose **File > Load from URL...** and paste the link into the File URL input field. Click **OK**.
6. Repeat for the remaining links.

If you ran an analysis and there are **no Data > IGV Visualization Files** links, the analysis generated multiple alignment BAM files over 10 GB, but did **not** consolidate the files. Click the **Launch BAM Consolidation** button to consolidate them.

IGV Visualization Files

No IGV Visualization Files were found

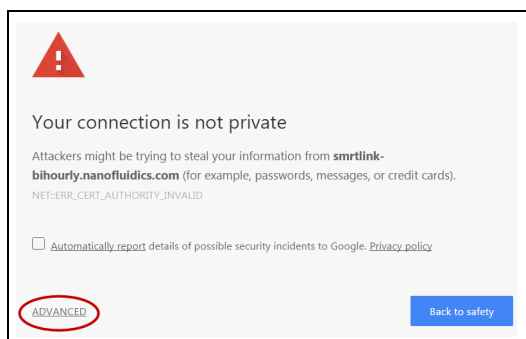
Launch BAM Consolidation

File	Path	Size	Type
Variants VCF	http://ip-172-32-0-82.us-west-1.compute.internal:9090/job-data/pb_sat/8f517e88-d47e-4250-beda-f6f04ea57894/call-pl_resequencing/job_resequencing/Sat2020a8-8a8f-4f63-b77f-6a4d490cfd6d/call-consensus/consensusu/8627b248-da29-433a-9a35-3f9474052dc/call-gather_vcf/execution/variants.vcf	583 bytes	vcf

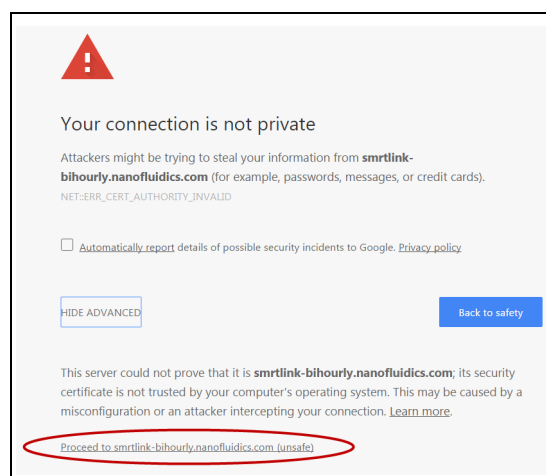
Using the PacBio® self-signed SSL certificate

SMRT Link v11.0 ships with a PacBio self-signed SSL certificate. If this is used at your site, security messages display when you try to login to SMRT Link for the **first time** using the Chrome browser. These messages may also display **other times** when accessing SMRT Link.

1. The first time you start SMRT Link after installation, you see the following. Click the **Advanced** link.



2. Click the **Proceed...** link. (You may need to scroll down.)



3. Close the window by clicking the **Close** box in the corner.



The **Login** dialog displays, where you enter the User Name and Password. The next time you access SMRT Link, the Login dialog displays **directly**.

Sequel® II system and Sequel IIe system output files

This section describes the data generated by the Sequel IIe system and Sequel II system for each SMRT Cell transferred to network storage.

Sequel IIe system output files

Following is a sample of the file and directory structure output by the **Sequel IIe** system (**not** including low-quality reads):

```
<your_specified_output_directory>/r64012_211206ee_183753/1_A01/
|--m64012ee_211206_183753.baz2bam_1.log
|--m64012ee_211206_183753.ccs.log
|--m64012ee_211206_183753.ccs_reports.json
|--m64012ee_211206_183753.ccs_reports.txt
|--m64012ee_211206_183753.consensusreadset.xml
|--m64012ee_211206_183753.hifi_reads.bam
|--m64012ee_211206_183753.hifi_reads.bam.pbi
|--m64012ee_211206_183753.sts.xml
|--m64012ee_211206_183753.zmw_metrics.json.gz
|--m64012ee_211206_183753.transferdone
```

If **5mC CpG Detection** is performed, the following additional files are output:

```
|-- m64012ee_211206_183753.5mc_report.json
|-- m64012ee_211206_183753.primrose.log
```

If **on-instrument demultiplexing** is performed, the following additional files are output. **Note:** The undemultiplexed `hifi_reads.bam` file is **not** transferred; it is partitioned into the file structure shown below:

```
|-- bc1001--bc1001/m64012e_211206_183753.bc1001--bc1001.consensusreadset.xml
|-- bc1001--bc1001/m64012e_211206_183753.hifi_reads.bc1001--bc1001.bam
|-- bc1001--bc1001/m64012e_211206_183753.hifi_reads.bc1001--bc1001.bam.pbi
|-- m64012e_211206_183753.barcodes.fasta
|-- m64012e_211206_183753.lima.log
|-- m64012e_211206_183753.lima_counts.txt
|-- m64012e_211206_183753.lima_guess.json
|-- m64012e_211206_183753.lima_guess.txt
|-- m64012e_211206_183753.lima_reports.txt
|-- m64012e_211206_183753.lima_summary.txt
|-- m64012e_211206_183753.unbarcoded.consensusreadset.xml
|-- m64012e_211206_183753.unbarcoded.hifi_reads.bam
|-- m64012e_211206_183753.unbarcoded.hifi_reads.bam.pbi
```

In these examples:

- `r64012ee_211206_183753` is a directory containing the output files associated with **one** run.
- `r64012ee` is the instrument ID number.

The run directory includes a subdirectory for each collection/cell associated with a sample well - in this case 1_A01. The collection/cell subdirectory can include the following output files:

- `ccs.log`: Log file from the CCS analysis. Informative for debugging and performance tracking by PacBio.
- `ccs_reports.json`, `ccs_reports.txt`: Contains processing metrics summarizing how many ZMWs generated HiFi reads, and how many ZMWs failed to generate CCS reads. These files contain the same information, and are used internally by PacBio Technical Support.
- `hifi.reads.bam`: Contains the HiFi reads in BAM format.
Note: If low-quality reads are included in the Run Design, the Sequel IIe system will output a `reads.bam` file, which contains **HiFi reads and non-HiFi** reads:
 - HiFi reads (QV 20 or higher)
 - Lower-quality but still polished consensus reads (QV 1 - QV 20)
 - Unpolished consensus reads (RQ=-1)
 - 0- or 1-pass subreads unaltered (RQ=-1)

The `reads.bam` file should **not** be used by itself as input for non-SMRT Link tools that expect \geq QV 20. The BAM format is a binary, compressed, record-oriented container format for raw or aligned sequence reads. The associated SAM format is a text representation of the same data. The BAM specifications are maintained by the SAM/BAM Format Specification Working Group. BAM files produced by **all** Sequel II systems are **fully compatible** with the BAM specification. For more information on the BAM file format specifications, click [here](#).

- `hifi.reads.bam.pbi`: Index file that allows for random access of HiFi reads in the BAM file.
- `sts.xml`: Contains summary statistics about the collection/cell and its post-processing.
- `zmw_metrics.json.gz`: Contains processing information used to generate RunQC plots.
- `5mc_report.json`, `primrose.log`: Contains information about 5mC CpG Detection analysis (using the `primrose` tool), if performed.
- `<Barcode Name>.consensusreadset.xml`: Contains reads associated with a specific barcode.
- `<Barcode Name>.bam`: Contains HiFi reads associated with a specific barcode, in .bam format.
- `<Barcode Name>.bam.pbi`: Index file that allows for random access of HiFi reads in the BAM file.
- `<Barcode Name>.fasta`: Contains reads associated with the specific barcode, in FASTA format.
- `lima.log`: Log file from the demultiplexing analysis, if performed. Informative for debugging and performance tracking by PacBio.
- `lima.counts.txt`: Contains the counts of each observed barcode pair. Only passing ZMWs are counted.

- `lima.guess.json`, `lima.guess.txt`: Describes the barcode subsetting process activated using the `--peek` and `--guess` options. These files contain the same information, and are used internally by PacBio Technical Support.
- `lima.reports.txt`: A tab-separated file describing each ZMW, unfiltered. This is useful information for investigating the demultiplexing process and the underlying data. A single row contains **all** reads from a single ZMW.
- `lima.summary.txt`: Lists how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- `unbarcoded.consensusreadset.xml`, `unbarcoded.hifi_reads.bam`, `unbarcoded.hifi_reads.bam.pbi`: Contains information on HiFi reads **not** associated with any barcode.

Note: The Sequel Ie system runs CCS on-instrument by default and the `subreads.bam`, `subreads.bam.pbi`, `scraps.bam` and `scraps.bam.pbi` files are no longer generated and are **not** available. Even though the `subreads.bam` and `subreads.bam.pbi` files are **not** accessible by default, there is a mechanism available to enable their output. For detailed instructions on how to enable the output of these files, contact your Field Applications Support team members.

HiFi reads generation

A standard Run Design performs on-instrument CCS analysis **without** including low-quality reads and generates a `hifi_reads.bam` file and transfers it to the network server. If low-quality reads are included in the Run Design, the Sequel Ie system will produce a `reads.bam` file, which contains HiFi reads and non-HiFi reads, and should **not** be used unfiltered as input for tools that expect \geq QV 20. SMRT Link **automatically** launches an **Export Reads** job on the `reads.bam` to filter out the HiFi reads, and generates the following HiFi data files by default:

- `<Movie_Name>.hifi_reads.fastq.gz` - Gzipped FASTQ file containing HiFi reads.
- `<Movie_Name>.hifi_reads.fasta.gz` - Gzipped FASTA file containing HiFi reads.
- `<Movie_Name>.hifi_reads.bam` - BAM file containing HiFi reads.

If **not** using SMRT Link for subsequent analysis, use these three files as input with any third-party analysis tools.

Finding the `hifi_reads` files generated using On-Instrument CCS

1. In Run QC, click the desired run, then click the sample name to view the CCS Data Set.
2. Click **Analyses** in the left-side panel.
3. Click the **Export Reads** analysis.

[Dataset Overview](#)

[Adapter Report](#)

[Raw Data Report](#)

[CCS Analysis Report](#)

Completed Analyses

Name	State	Id	Date Created	Created By	Analysis Application
Auto Analyses of 64009 0211 SAT OICCS	SUCCESSFUL	27671	2021-02-11, 01:45:49 PM	aswei	
Export Reads of 2kb Lambda OICCS	SUCCESSFUL	27672	2021-02-11, 01:45:49 PM	aswei	Export Reads

4. To locate the directory containing the three `hifi_reads` files, append `/outputs` to the path shown.

▼ Analysis Overview

Status

Display All

▶ Data

Analysis

Export Reads of 2kb Lambda OICCS

Analysis ID

27672

From Multi-Job

27671

Status

SUCCESSFUL: 5 tasks finished

Created By

aswei

Date Created

2021-02-11, 01:45:49 PM

Date Updated

2021-02-11, 08:34:01 PM

Application

Export Reads

SMRT Link Version

10.1.0.115913

Inputs

Data Type	Name	Import Complete
ConsensusReadSet	HIFI Reads: 2kb Lambda OICCS-Cell1 [...]	Yes

Path

/jpbi/dept/secondary/siv/smartlink/smartlink-siv-alpha/smartlink_5.1.0.SNAPSHOT13617/userdata/jobs_root/0000/0000027/0000027672

Sequel II system output files

Following is a sample of the file and directory structure output by the Sequel II system:

```
<your_specified_output_directory>/r64008_20160116_003347/1_A01
|-- m64008_160116_003634.baz2bam_1.log
|-- m64008_160116_003634.scrap.bam
|-- m64008_160116_003634.scrap.bam.pbi
|-- m64008_160116_003634.subreads.bam
|-- m64008_160116_003634.subreads.bam.pbi
|-- m64008_160116_003634.subreadset.xml
|-- m64008_160116_003634.sts.xml
|-- m64008_160116_003634.transferdone
```

Files output by the Sequel II system include:

- `scrap.bam` and `scrap.bam.pbi`: These files contain sequence data outside of the high-quality region, rejected subreads, excised adapter and possible barcode sequences, as well as spike-in control sequences. (The basecaller marks regions of single molecule sequence activity as high-quality.) **Note**: This applies to files generated by Sequel Instrument Control Software (ICS) v3.1.0 or later.
- `subreads.bam`: The Sequel II system output **one** `subreads.bam` file per collection/cell, which contains unaligned base calls from high-quality regions. This file is transferred from the instrument to network storage, then is used as **input** for secondary analysis by PacBio's SMRT Analysis software. Data in a `subreads.bam` file is analysis-ready; all of the data present should be quality-filtered for analyses. Subreads that contain information such as double-adapter inserts or single-molecule artifacts are **not** used in secondary analysis, and are excluded from this file and placed in `scrap.bam`.

- `subreads.bam.pbi`: Provides backwards-compatibility with the APIs enabled for accessing the `cmp.h5` file.
- `subreadset.xml`: This file is needed to import data into SMRT Link.
- `sts.xml`: Contains summary statistics about the collection/cell and its post-processing.
- `transferdone`: Contains a list of files successfully transferred.

Frequently asked questions

What are the minimum files needed to analyze data on SMRT Link?

- `.bam` file
- `bam.pbi` file
- `subreadset.xml` or `consensusreadset.xml` file

What is the average size of the file bundle for a 30-hour movie of HiFi reads?

Approximately 50 Gb.

What is the difference between a regular `.bam` file and an `aligned.bam` file?

The `subreads.bam` file contains all the subreads sequences, while the `aligned.bam` file additionally contains the genomic coordinates of the reads mapped to a reference sequence.

The `subreads.bam` file is created by the Sequel II systems, while the `aligned.bam` file is created by SMRT Link after running mapping analysis applications.

Secondary analysis output files

This is data produced by secondary analysis, which is performed on the primary analysis data generated by the instrument.

- All files for a specific job reside in **one** directory named according to the job ID number.
- Every job result has the following file structure. **Example:**

```
$SMRT_ROOT/userdata/jobs_root/0000/00000000/00000000002/
├── cromwell-job -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│   └── pb_demux_subreads_auto/24e691c8-8d0d-4670-9db3-c7cb1126e8f8
│       ├── entry-points
│       │   └── ae6f1c2c-b4a2-41cc-8e44-98b494f12a57.subreadset.xml
│       ├── logs
│       │   ├── pb_simple_mapping
│       │   │   └── 24e691c8-8d0d-4670-9db3-c7cb1126e8f8
│       │   │       ├── call-mapping
│       │   │       │   └── execution
│       │   │       │       ├── stderr
│       │   │       │       └── stdout
│       │   └── workflow.24e691c8-8d0d-4670-9db3-c7cb1126e8f8.log
│       ├── outputs
│       │   ├── mapping.report.json -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│       │   │   └── pb_simple_mapping/24e691c8-8d0d-4670-9db3-c7cb1126e8f8/call-mapping/execution/
│       │   │       mapping.report.json
│       │   └── mapped.bam -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│       │       └── pb_simple_mapping/24e691c8-8d0d-4670-9db3-c7cb1126e8f8/call-mapping/execution/
│       │           mapped.bam
│       ├── pbscala-job.stderr
│       ├── pbscala-job.stdout
│       └── workflow
│           ├── analysis-options.json
│           ├── datastore.json
│           ├── engine-options.json
│           ├── inputs.json
│           ├── metadata.json
│           ├── metadata-summary.json
│           ├── task-timings.metadata.json
│           └── timing-diagram.html
```

- `logs/`: Contains log files for the job.
 - `workflow.<UUID>.log`: Global log of each significant step in the job and snippets from a task's `stderr` output if the job failed.
 - The same directory contains `stdout` and `stderr` for individual tasks.
- `cromwell-job/`: Symbolic link to the actual Cromwell execution directory, which resides in another part of the `jobs-root` directory. Contains subdirectories for each workflow task, along with executable scripts, output files, and `stderr/stdout` for the task.
 - `call-tool_name/execution/`: Example of an individual task directory (This is replaced with `<task_id>` below.)
 - `<task_id>/stdout`: General task `stdout` log collection.
 - `<task_id>/stderr`: General task `stderr` log collection.

- `<task_id>/script`: The SMRT Tools command for the given analysis task.
- `<task_id>/script.submit`: The JMS submission script wrapping `run.sh`.
- `<task_id>stdout.submit`: The `stdout` collection for the `script.submit` script.
- `<task_id>/stderr.submit`: The `stderr` collection for the `script.submit` script.
- `workflow/`: Contains JSON files for job settings and workflow diagrams.
 - `datastore.json`: JSON file representing all output files imported by SMRT Link.
- `outputs/`: A directory containing symbolic links to all datastore files, which reside in the Cromwell execution directory. This is provided as a convenience and is **not** intended as a stable API; note that external resources from dataset XML and report JSON file are **not** included here. Demultiplexing outputs are nested in additional subdirectories.
- `pbscala-job.stderr`: Log collection of `stderr` output from the SMRT Link job manager.
- `pbscala-job.stdout`: Log collection of `stdout` output from the SMRT Link job manager. (**Note**: This is the file displayed as **Data > SMRT Link Log** on the analysis results page.)

A SMRT Link job generates several types of output files. You can use these data files as input for further processing, pass on to collaborators, or upload to public genome sites. Depending on the analysis application being used, the `output` directory contain files in the following formats:

- **BAM**: Binary version of the Sequence Alignment Map (SAM) format. (See [here](#) for details.)
- **BAI**: The `samtools` index file for a file generated in the BAM format.
- **BED**: Format that defines the data lines displayed in an annotation track. (See [here](#) for details.)
- **CSV**: Comma-Separated Values file. Can be viewed using Microsoft Excel or a text editor.
- **FASTA/FASTQ**: Sequence files that contains either nucleic acid sequence (such as DNA) or protein sequence information. FASTA/Q files store multiple sequences in a single file. FASTQ files also include per-base quality scores. (See [here](#) or [here](#) for details.)
- **GFF**: General Feature Format, used for describing genes and other features associated with DNA, RNA and protein sequences. (See [here](#) for details.)
- **PBI**: PacBio index file. (This is a PacBio-specific file type.)
- **VCF**: Variant Call Format, for use with the molecular visualization and analysis program VMD. (See [here](#) for details.)

To download data files created by SMRT Link:

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. Click the job link of interest.
3. Click **Data > File Downloads**, then click the appropriate file. The file is downloaded according to your browser settings.
 - **(Optional)** Click the small icon to the right of the file name to copy the file's path to the Clipboard.

Configuration and user management

LDAP

SMRT Link supports the use of LDAP for user login and authentication. **Without** LDAP integration with SMRT Link, only **one** user (with the login `admin/admin`) is enabled. You can add new users **after** SMRT Link is integrated and configured to work with LDAP; you can also add new users using WSO2 API Manager or Keycloak **without** LDAP integration.

- For details on integrating LDAP and SMRT Link, see the document **SMRT Link software installation guide (v11.0)**.

SSL

SMRT Link requires the use of Secure Sockets Layer (SSL) to enable access via HTTP over SSL (HTTPS), so that SMRT Link logins and data are encrypted during transport to and from SMRT Link. SMRT Link includes an Identity Server (WSO2 API Manager or Keycloak), which can be configured to integrate with your LDAP/AD servers and enable user authentication using your organizations' user name and password. To ensure a secure connection between the SMRT Link server and your browser, the SSL certificate can be installed **after** completing SMRT Link installation.

It is important to note that PacBio will **not** provide a signed SSL certificate, however – once your site has obtained one – PacBio tools can be used to install it and configure SMRT Link to use it. You will need a certificate issued by a Certificate Authority (CA, sometimes referred to as a **certification authority**). PacBio has tested SMRT Link with certificates from the following certificate vendors: VeriSign, Thawte and digicert.

Note: PacBio recommends that you consult your IT administrator about obtaining an SSL certificate.

Alternatively, you can use your site's self-signed certificate.

SMRT Link ships with a PacBio self-signed SSL certificate. If used, **each** user will need to accept the browser warnings related to access in an insecure environment. Otherwise, your IT administrator can configure desktops to **always** trust the provided self-signed certificate. Note that SMRT Link is installed within your organization's secure network, behind your organization's firewall.

- For details on updating SMRT Link to use an SSL certificate, see the document **SMRT Link software installation guide (v11.0)**.

The following procedures are available **only** for SMRT Link users whose role is **Admin**.

Adding and deleting SMRT Link users

1. Choose **Gear > Configure**, then click **User Management**.
2. There are two ways to find users:
 - To display **all** SMRT Link users: Click **Display all Enabled Users**.
 - To find a specific user: Enter a user name, or partial name, and click **Search By Name**.
3. Click the desired user. If the user status is **Enabled**, the user has access to SMRT Link; **Disabled** means the user **cannot** access SMRT Link.
 - To **add** a SMRT Link user: Click the **Enabled** button, then assign a role. (See below for details.)
 - To **disable** a SMRT Link user: Click the **Disabled** button.
4. Click **Save**.

Assigning user roles

SMRT Link supports three user roles: **Admin**, **Lab Tech**, and **Bioinformatician**. Roles define which SMRT Link modules a user can access. The following table lists the privileges associated with the three user roles:

Tasks/privileges	Admin	Lab Tech	Bioinformatician
Add/delete SMRT Link users	Y	N	N
Assign roles to SMRT Link users	Y	N	N
Update SMRT Link software	Y	N	N
Access Sample Setup module	Y	Y	N
Access Run Design module	Y	Y	N
Access Run QC module	Y	Y	Y
Access Data Management module	Y	Y	Y
Access SMRT Analysis module	Y	Y	Y

1. Choose **Gear > Configure**, then click **User Management**.
2. There are two ways to find users:
 - To display **all** SMRT Link users: Click **Display all Enabled Users**.
 - To find a specific user: Enter a user name, or partial name, and click **Search By Name**.
3. Click the desired user.
4. Click the **Role** field and select one of the three roles. (A **blank** role means that this user **cannot** access SMRT Link.)

- **Note:** There can be **multiple** users with the Admin role; but there **must** always be at least **one** Admin user.
5. Click **Save**.

Hardware/software requirements

Client hardware requirements

- SMRT Link requires a minimum screen resolution of 1600 by 900 pixels.

Client software requirements

- SMRT Link **requires** the Google® Chrome web browser, version 90 or later.

Note: SMRT Link **server** hardware and software requirement are listed in the document **SMRT Link software installation guide (v11.0)**.

Appendix A - PacBio terminology

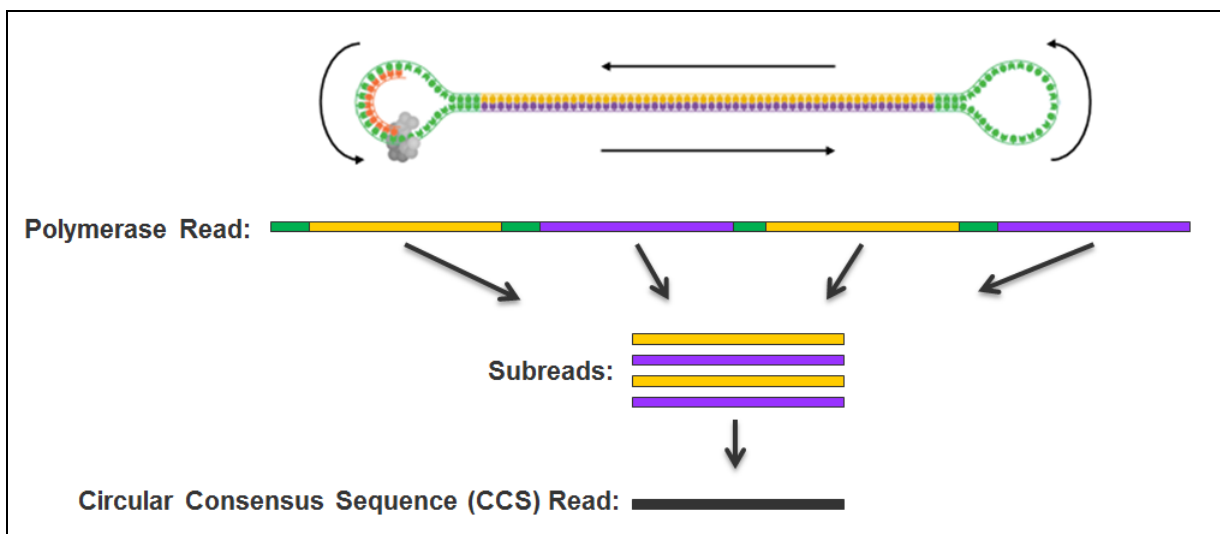
General terminology

- **SMRT[®] Cell**: Consumable substrates comprising arrays of zero-mode waveguide nanostructures. SMRT Cells are used in conjunction with the DNA sequencing kit for on-instrument DNA sequencing.
- **SMRTbell[®] template**: A double-stranded DNA template capped by hairpin adapters (i.e., SMRTbell adapters) at both ends. A SMRTbell template is topologically circular and structurally linear, and is the library format created by the DNA template prep kit.
- **collection**: The set of data collected during real-time observation of the SMRT Cell; including spectral information and temporal information used to determine a read.
- **Zero-mode waveguide (ZMW)**: A nanophotonic device for confining light to a small observation volume. This can be, for example, a small hole in a conductive layer whose diameter is too small to permit the propagation of light in the wavelength range used for detection. Physically part of a SMRT Cell.
- **Run Design**: Specifies
 - The samples, reagents, and SMRT Cells to include in the sequencing run.
 - The run parameters such as movie time and loading to use for the sample.
- **adaptive loading**: Uses active monitoring of the ZMW loading process to predict a favorable loading end point.
- **unique molecular yield**: The sum total length of unique single molecules that were sequenced. It is calculated as the sum of per-ZMW median subread lengths.

Read terminology

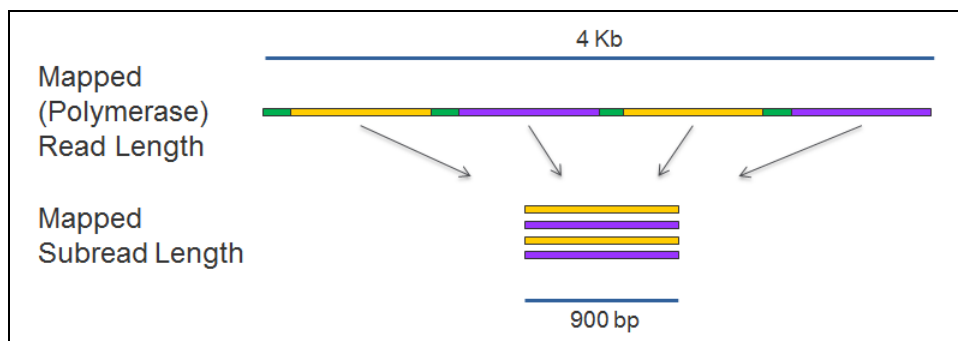
- **polymerase read**: A sequence of nucleotides incorporated by the DNA polymerase while reading a template, such as a circular SMRTbell template. They can include sequences from adapters and from one or multiple passes around a circular template, which includes the insert of interest. Polymerase reads are most useful for quality control of the instrument run. Polymerase read metrics primarily reflect movie length and other run parameters rather than insert size distribution. Polymerase reads are trimmed to include only the high-quality region. **Note**: Sample quality is a major factor in polymerase read metrics.
- **subreads**: Each polymerase read is partitioned to form one or more subreads, which contain sequence from a single pass of a polymerase on a single strand of an insert within a SMRTbell template and no adapter sequences. The subreads contain the full set of quality values and kinetic measurements. Subreads are useful for applications such as *de novo* assembly, base modification analysis, and so on.
- **longest subread length**: The mean of the maximum subread length per ZMW.

- **insert length:** The length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters.
- **circular consensus (CCS) reads:** The consensus sequence resulting from alignment between subreads taken from a single ZMW. Generating CCS reads does **not** include or require alignment against a reference sequence but **does** require at least two full-pass subreads from the insert. CCS reads are generated with CCS analysis. CCS reads with quality value equal to or greater than 20 are called **HiFi reads**.
- **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.



Read length terminology

- **mapped polymerase read length:** Approximates the sequence produced by a polymerase in a ZMW. The total number of bases along a read from the first adapter of aligned subread to the last adapter or aligned subread.
- **mapped subread length:** The length of the subread alignment to a target reference sequence. This does **not** include the adapter sequence.



Secondary analysis terminology

- **secondary analysis:** Follows primary analysis and uses basecalled data. It is application-specific, and may include:

- Filtering/selection of data that meets a desired criteria, such as quality, read length, and so on.
- Comparison of reads to a reference or between each other for mapping and variant calling, consensus sequence determination, alignment and assembly (*de novo* or reference-based), variant identification, and so on.
- Quality evaluations for a sequencing run, consensus sequence, assembly, and so on.
- PacBio's SMRT Analysis contains a variety of secondary analysis applications including RNA and Epigenomics analysis tools.
- **secondary analysis application:** A secondary analysis workflow that may include multiple analysis steps. Examples include *de novo* assembly, RNA and epigenomics analysis.
- **consensus:** Generation of a consensus sequence from multiple-sequence alignment.
- **filtering:** Removes reads that do not meet the Read Length criteria set by the user.
- **mapping:** Local alignment of a read or subread to a reference sequence.
- **Auto Analysis:** Allows a specific analysis to be automatically run after a sequencing run has finished and the data is transferred to the SMRT Link server. The analysis can include demultiplexed outputs.
 - Auto Analysis works with **all** Sequel II systems.
- **Pre Analysis:** The process of CCS analysis and/or demultiplexing of Sequel basecalled data. Pre Analysis occurs **before** Auto Analysis.
 - Pre Analysis works with **all** Sequel II systems.

Accuracy terminology

- **circular consensus accuracy:** Accuracy based on consensus sequence from multiple sequencing passes around a single circular template molecule.
- **consensus accuracy:** Accuracy based on aligning multiple sequencing reads or subreads together.
- **polymerase read quality:** A trained prediction of a read's mapped accuracy based on its pulse and base file characteristics (peak signal-to-noise ratio, inter-pulse distance, and so on).

Appendix B - Data search

Use this function to search for jobs, Data Sets, barcode files, or reference files.

To search the entire table

1. Enter a text query into the Search box. This searches **every field** in the table, and displays **all** table rows containing the search characters.

To search for a value within a column

1. Click the small filter icon at the right of the column name.
2. Enter a value; **all** table rows meeting the search criteria display. (To select a **different** search operator, click the droplist and select another search operator. Different search operators are available, based on the column's data type.)

- For the **Analysis State** column only, click one or more of the job states of interest: **Select All, Created, Running, Submitted, Terminated, Successful, Failed, or Aborted.**
- For **Date fields** only, click the small calendar and select a date.

Numeric field operators

- Equals, Not equal
- Greater than, Greater than or equals

- Less than, Less than or equals
- In range

Text field operators

- Contains, Not contains
- Equals, Not equal
- Starts with, Ends with

Date field operators

- Equals, Not equal
- Greater than, Less than
- In range

Appendix C - BED file format for Target Regions report

With the HiFi Mapping application, an optional **Target Regions** report can be generated that displays the number (and percentage) of reads and subreads that hit specified target regions.

The BED file required to generate the Target Regions report includes the following fields; with one entry per line:

1. **chrom**: The name of the chromosome (such as `chr3`, `chrY`, `chr2_random`) or scaffold (such as `scaffold10671`).
2. **chromStart**: The starting position of the feature in the chromosome or scaffold. The first base in a chromosome is numbered 0.
3. **chromEnd**: The ending position of the feature in the chromosome or scaffold. The **chromEnd** base is **not** included in the display of the feature, however, the number in position format is represented. For example, the first 100 bases of chromosome 1 are defined as `chrom=1, chromStart=0, chromEnd=100`, and span the bases numbered 0-99 (**not** 0-100), but will represent the position notation `chr1:1-100`.
4. **(Optional) Region Name**.

Example: `lambda_NEB3011 15000 25000 Region2`

- Fields can be space- or tab-delimited.
- See [here](#) for details of the BED format.
- For details on the BED format's counting system, see [here](#) and [here](#).

Appendix D - Additional information included in the CCS Data Set Export report

When you export a Data Set and select **Export PDF Reports**, a report is produced which includes additional fields, listed below.

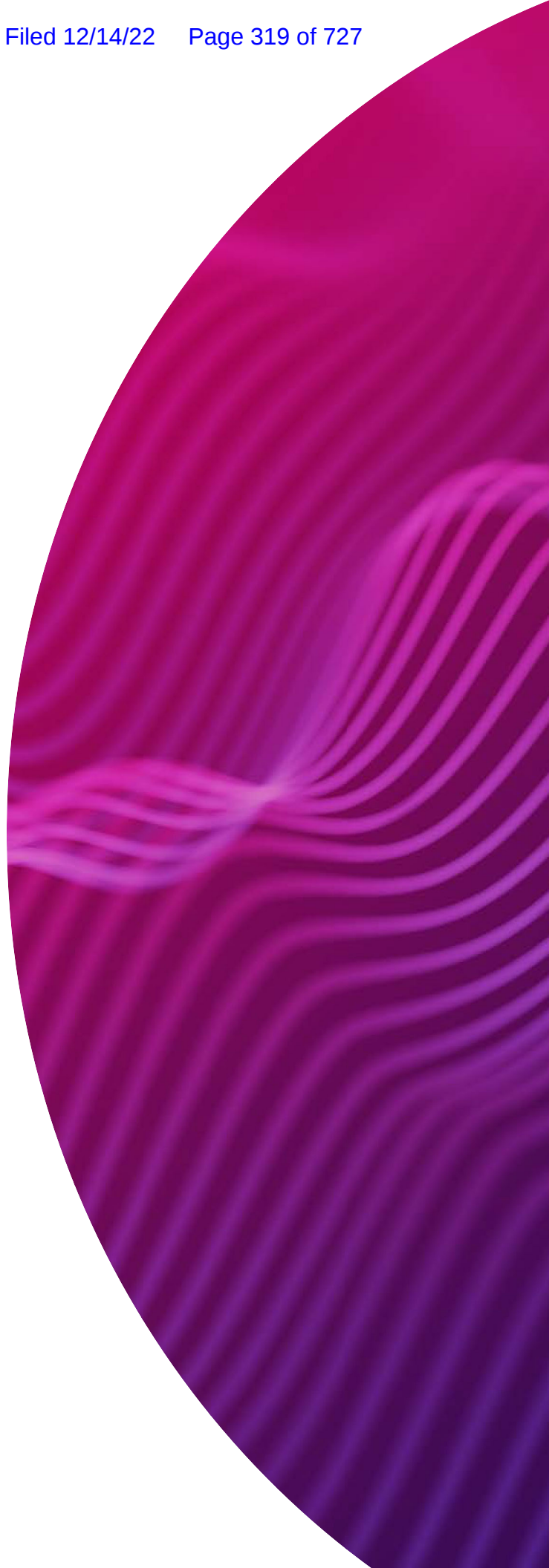
- See [“Exporting sequence, reference and barcode data” on page 42](#) for details on exporting Data Sets.
- The other fields and plots in this report are described in the appropriate Reports sections of [“PacBio® secondary analysis applications” on page 54](#).
- **ZMWs input:** The total number of ZMWs used as input in the Data Set.
- **ZMWs pass filters:** The number of ZMWs that passed **all** the filters.
- **ZMWs fail filters:** The number of ZMWs that failed **any** of the filters.
- **ZMWs shortcut filters:** The number of low-pass ZMWs skipped using the `--all` filter.
- **ZMWs with tandem repeats:** The number of ZMWs that did not generate CCS reads due to repeats larger than `--min-tandem-repeat-length`.
- **Below SNR threshold:** The number of ZMWs that did not generate CCS reads due to SNR below `--min-snr`.
- **Median length filter:** The number of ZMWs that did not generate CCS reads due to subreads that are <50% or >200% of the median subread length.
- **Lacking full passes:** The number of ZMWs that did not generate CCS reads due to having fewer than `--min-passes` full-length subreads.
- **Heteroduplex insertions:** The number of ZMWs that did not generate CCS reads due to single-strand artifacts.
- **Coverage drops:** The number of ZMWs that did not generate CCS reads due to coverage drops that would lead to unreliable polishing results.
- **Insufficient draft cov:** The number of ZMWs that did not generate CCS reads due to not having enough subreads aligned to the draft sequence end-to-end.
- **Draft too different:** The number of ZMWs that did not generate CCS reads due to having fewer than `--min-passes` full-length reads aligned to the draft sequence.
- **Draft generation error:** The number of ZMWs that did not generate CCS reads due to subreads that don't agree enough to generate a draft sequence.
- **Draft above --max-length:** The number of ZMWs that did not generate CCS reads due to a draft sequence longer than `--max-length`.
- **Draft below --min-length:** The number of ZMWs that did not generate CCS reads due to a draft sequence shorter than `--min-length`.
- **Reads failed polishing:** The number of ZMWs that did not generate CCS reads due to too many subreads dropped while polishing.

- **Empty coverage windows:** The number of ZMWs that did not generate CCS reads because at least one window had no coverage.
- **CCS did not converge:** The number of ZMWs that did not generate CCS reads because the draft sequence had too many errors that could not be polished in time.
- **CCS below minimum RQ:** The number of ZMWs that did not generate CCS reads because the predicted accuracy is below `--min-rq`.
- **Unknown error:** The number of ZMWs that did not generate CCS reads due to rare implementation errors.

EXHIBIT J



SMRT[®] Tools reference guide (v11.0)



Research use only. Not for use in diagnostic procedures.

P/N 102-278-500 Version 01 (April 2022)

© 2022, PacBio. All rights reserved.

Information in this document is subject to change without notice. PacBio assumes no responsibility for any errors or omissions in this document.

Certain notices, terms, conditions and/or use restrictions may pertain to your use of PacBio products and/or third party products. Refer to the applicable PacBio terms and conditions of sale and to the applicable license terms at <http://www.pacificbiosciences.com/licenses.html>.

Trademarks:

Pacific Biosciences, the PacBio logo, PacBio, Circulomics, Omnione, SMRT, SMRTbell, Iso-Seq, Sequel, Nanobind, and SBB are trademarks of Pacific Biosciences of California Inc. (PacBio). All other trademarks are the sole property of their respective owners.

See <https://github.com/broadinstitute/cromwell/blob/develop/LICENSE.txt> for Cromwell redistribution information.

PacBio

1305 O'Brien Drive

Menlo Park, CA 94025

www.pacb.com



SMRT® Tools reference guide (v11.0)

Introduction

This document describes the command-line tools included with SMRT® Link v11.0. These tools are for use by bioinformaticians working with secondary analysis results.

- The command-line tools are located in the `$SMRT_ROOT/smrtlink/smrtcmds/bin` subdirectory.

Installation

The command-line tools are installed as an integral component of the SMRT Link software. For installation details, see **SMRT Link software installation guide (v11.0)**.

- **Note:** SMRT Link v11.0 is for use with Sequel® II systems and Sequel IIe systems **only**.
- To install **only** the command-line tools, use the `--smrttools-only` option with the installation command, whether for a new installation or an upgrade. Examples:

```
smrtlink-*.run --rootdir smrtlink --smrttools-only
smrtlink-*.run --rootdir smrtlink --smrttools-only --upgrade
```

Supported chemistry

SMRT Link v11.0 supports all chemistry versions for **Sequel II systems**.

PacBio command-line tools

Following is information on the PacBio-supplied command-line tools included in the installation. For information on third-party tools installed, see [“Appendix B - Third party command-line tools” on page 113](#).

Tool	Description
bam2fasta/ bam2fastq	Converts PacBio® BAM files into gzipped FASTA and FASTQ files. See “bam2fasta/bam2fastq” on page 3 .
bamsieve	Generates a subset of a BAM or PacBio Data Set file based on either a list of hole numbers, or a percentage of reads to be randomly selected. See “bamsieve” on page 4 .
ccs	Calculates consensus sequences from multiple “passes” around a circularized single DNA molecule (SMRTbell® template). See “ccs” on page 7 .
dataset	Creates, opens, manipulates and writes Data Set XML files. See “dataset” on page 16 .
Demultiplex Barcodes	Identifies barcode sequences in PacBio single-molecule sequencing data. See “Demultiplex Barcodes” on page 23 .

Tool	Description
export-datasets	Takes one or more PacBio dataset XML files and packages all contents into a single ZIP archive. See “export-datasets” on page 35 .
export-job	Takes one SMRT Link Analysis job and packages all contents into a single ZIP archive. See “export-job” on page 36 .
gcpp	Variant-calling tool which provides several variant-calling algorithms for PacBio sequencing data. See “gcpp” on page 38 .
Genome Assembly	Generates <i>de novo</i> assemblies using HiFi reads. See “Genome Assembly” on page 41 .
HiFiViral SARS-CoV-2 Analysis	Analyzes multiplexed viral surveillance samples for SARS-CoV-2. See “HiFiViral SARS-CoV-2 Analysis” on page 49 .
ipdSummary	Detects DNA base-modifications from kinetic signatures. See “ipdSummary” on page 52 .
isoseq3	Characterizes full-length transcripts and generates full-length transcript isoforms, eliminating the need for computational reconstruction. See “isoseq3” on page 56 .
juliet	A general-purpose minor variant caller that identifies and phases minor single nucleotide substitution variants in complex populations. See “juliet” on page 60 .
Microbial Genome Analysis	Generates <i>de novo</i> assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb and performs base modification detection, using HiFi reads. See “Microbial Genome Analysis” on page 68 .
motifMaker	Identifies motifs associated with DNA modifications in prokaryotic genomes. See “motifMaker” on page 73 .
pbcromwell	PacBio’s wrapper for the <code>cromwell</code> scientific workflow engine used to power SMRT Link. For details on how to use <code>pbcromwell</code> to run workflows, see “pbcromwell” on page 75 .
pbindex	Creates an index file that enables random access to PacBio-specific data in BAM files. See “pbindex” on page 80 .
pbmarkdup	Marks or removes duplicates reads from CCS reads. See “pbmarkdup” on page 81 .
pbbmm2	Aligns PacBio reads to reference sequences; a SMRT wrapper for <code>minimap2</code> . See “pbbmm2” on page 84 .
pbservice	Performs a variety of useful tasks within SMRT Link. See “pbservice” on page 91 .
pbsv	Structural variant caller for PacBio reads. See “pbsv” on page 96 .
pbvalidate	Validates that files produced by PacBio software are compliant with PacBio’s own internal specifications. See “pbvalidate” on page 100 .
primrose	Performs motif-calling to detect 5mC CpG sites in HiFi BAM files. See “primrose” on page 103 for details.
runqc-reports	Generates Run QC reports. See “runqc-reports” on page 104 .
summarizeModifications	Generates a GFF summary file from the output of base modification analysis combined with the coverage summary GFF generated by resequencing pipelines. See “summarize Modifications” on page 105 .

**bam2fasta/
bam2fastq**

The `bam2fasta` and `bam2fastq` tools convert PacBio BAM or Data Set files into gzipped FASTA and FASTQ files, including demultiplexing of barcoded data.

Usage

Both tools have an identical interface and take BAM and/or Data Set files as input.

```
bam2fasta [options] <input>
bam2fastq [options] <input>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-o, --output</code>	Specifies the prefix of the output file names. <code>-</code> implies streaming. Note: Streaming is not supported with compression or with the <code>split_barcodes</code> option.
<code>-c</code>	Specifies the Gzip compression level. Values are <code>[1, 2, 3, 4, 5, 6, 7, 8, 9]</code> . (Default = 1)
<code>-u</code>	Specifies that the output FASTA/FASTQ files are not compressed. <code>.gz</code> is not added to the output file names, and <code>-c</code> settings are ignored.
<code>--split-barcodes</code>	Splits the output into multiple FASTA/FASTQ files, by barcode pairs. Note: The <code>bam2fasta/bam2fastq</code> tools inspect the <code>bc</code> tag in the BAM file to determine the 0-based barcode indices from their respective positions in the barcode FASTA file.
<code>-p, --seqid-prefix</code>	Specifies the prefix for the sequence IDs used in the FAST/FASTQ file headers.

Examples

```
bam2fasta -o projectName m54008_160330_053509.subreads.bam
```

```
bam2fastq -o myEcoliRuns m54008_160330_053509.subreads.bam
m54008_160331_235636.subreads.bam
```

```
bam2fasta -o myHumanGenome m54012_160401_000001.subreadset.xml
```

Input files

- One or more `*.bam` files
- `*.subreadset.xml` file (Data Set file)

Output files

- `*.fasta.gz`
- `*.fastq.gz`

bamsieve The `bamsieve` tool creates a subset of a BAM or PacBio Data Set file based on either a list of hole numbers to include or exclude, or a percentage of reads to be randomly selected, while keeping all subreads within a read together. Although `bamsieve` is BAM-centric, it has some support for dataset XML and will propagate metadata, as well as scraps BAM files in the special case of SubreadSets. `bamsieve` is useful for generating minimal test Data Sets containing a handful of reads.

`bamsieve` operates in two modes: **list** mode where the ZMWs to keep or discard are explicitly specified, or **percentage/count** mode, where a fraction of the ZMWs is randomly selected.

ZMWs may be listed in one of several ways:

- As a comma-separated list on the command line.
- As a flat text file, one ZMW per line.
- As another PacBio BAM or Data Set of any type.

Usage

```
bamsieve [-h] [--version] [--log-file LOG_FILE]
          [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug | --quiet
          | -v]
          [--show-zmws] [--include INCLUDE LIST] [--exclude EXCLUDE LIST]
          [--percentage PERCENTAGE] [-n COUNT] [-s SEED]
          [--ignore-metadata] [--barcodes]
          input_bam [output_bam]
```

Required	Description
input_bam	The name of the input BAM file or Data Set from which reads will be read.
output_bam	The name of the output BAM file or Data Set where filtered reads will be written to. (Default = None)

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. (Default = None, writes to stdout.)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL]. (Default = WARNING)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
-v, --verbose	Sets the verbosity level. (Default = NONE)
--show-zmws	Prints a list of ZMWs and exits. (Default = False)
--include INCLUDE LIST	Specifies the ZMWs to include in the output. This can be a comma-separated list of ZMWs, or a file containing a list of ZMWs (one hole number per line), or a BAM/ Data Set file. (Default = NONE)

Options	Description
<code>--exclude EXCLUDE LIST</code>	Specifies the ZMWs to exclude from the output. This can be a comma-separated list of ZMWs, or a file containing a list of ZMWs (one hole number per line), or a BAM/Data Set file that specifies ZMWs. (Default = NONE)
<code>--percentage PERCENTAGE</code>	Specifies a percentage of a SMRT® Cell to recover (Range = 1-100) rather than a specific list of reads. (Default = NONE)
<code>-n COUNT, --count COUNT</code>	Specifies a specific number of ZMWs picked at random to recover. (Default = NONE)
<code>-s SEED, --seed SEED</code>	Specifies a random seed for selecting a percentage of reads. (Default = NONE)
<code>--ignore-metadata</code>	Discard the input Data Set metadata. (Default = False)
<code>--barcodes</code>	Specifies that the include/exclude list contains barcode indices instead of ZMW numbers. (Default = False)

Examples

Pulling out two ZMWs from a BAM file:

```
bamsieve --include 111111,222222 full.subreads.bam sample.subreads.bam
```

Pulling out two ZMWs from a Data Set file:

```
bamsieve --include 111111,222222 full.subreadset.xml sample.subreadset.xml
```

Using a text list:

```
bamsieve --include zmw.txt full.subreads.bam sample.subreads.bam
```

Using another BAM or Data Set as a list:

```
bamsieve --include mapped.alignmentset.xml full.subreads.bam mappable.subreads.bam
```

Generating a list of ZMWs from a Data Set:

```
bamsieve --show-zmws mapped.alignmentset.xml > mapped_zmws.txt
```

Anonymizing a Data Set:

```
bamsieve --include zmw.txt --ignore-metadata --anonymize full.subreadset.xml
anonymous_sample.subreadset.xml
```

Removing a read:

```
bamsieve --exclude 111111 full.subreadset.xml filtered.subreadset.xml
```

Selecting 0.1% of reads:

```
bamsieve --percentage 0.1 full.subreads.bam random_sample.subreads.bam
```

Selecting a different 0.1% of reads:

```
bamsieve --percentage 0.1 --seed 98765 full.subreads.bam random_sample.subreads.bam
```

Selecting just two ZMWs/reads at random:

```
bamsieve --count 2 full.subreads.bam two_reads.subreads.bam
```

Selecting by barcode:

```
bamsieve --barcodes --include 4,7 full.subreads.bam two_barcodes.subreads.bam
```

Generating a tiny BAM file that contains only mappable reads:

```
bamsieve --include mapped.subreads.bam full.subreads.bam mappable.subreads.bam  
bamsieve --count 4 mappable.subreads.bam tiny.subreads.bam
```

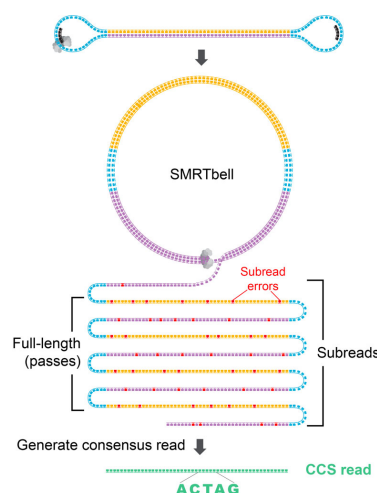
Splitting a Data Set into two halves:

```
bamsieve --percentage 50 full.subreadset.xml split.1of2.subreadset.xml  
bamsieve --exclude split.1of2.subreadset.xml full.subreadset.xml  
split.2of2.subreadset.xml
```

Extracting Unmapped Reads:

```
bamsieve --exclude mapped.alignmentset.xml movie.subreadset.xml unmapped.subreadset.xml
```

ccs Circular Consensus Sequencing (CCS) Analysis computes consensus sequences from multiple “passes” around a circularized single DNA molecule (SMRTbell® template). CCS analysis uses the Arrow framework to achieve optimal consensus results given the number of passes available.



CCS analysis workflow

1. Initial filtering

- Filter ZMWs: Remove ZMWs with signal-to-noise ratio (SNR) below `--min-snr`.
- Filter subreads: Remove subreads with lengths <50% or >200% of the median subread length. Stop if the number of full-length subreads is fewer than `--min-passes`.

2. Generate draft

- The polish stage iteratively improves upon a candidate template sequence. Because polishing is very compute-intensive, it is desirable to start with a template that is as close as possible to the true sequence of the molecule to reduce the number of iterations until convergence. The `ccs` software does **not** pick a full-length subread as the initial template to be polished, but instead generates an approximate draft consensus sequence using our improved implementation of the `sparc` graph consensus algorithm. This algorithm depends on a subread-to-backbone alignment that is generated by the `pancake` mapper developed by PacBio, using `edlib` as the core aligner. Typically, subreads have accuracy of around 90% and the draft consensus has a higher accuracy, but is still below 99%.
- Stop if the draft length is shorter than `--min-length` and longer than `--max-length`.

3. Alignment

- Align subreads to the draft consensus using `pancake` with KSW2 for downstream windowing and filtering.

4. Windowing

- Divide the subread-to-draft alignment into overlapping windows with a target size of 22 bp with ± 2 bp overlap. Avoid breaking windows at simple repeats (homopolymers to 4-mer repeats) to reduce edge cases at window borders. Windowing reduces the algorithm run time from quadratic to linear in the insert size.

5. Single-strand artifacts

- Identify heteroduplexes, where one strand of the SMRTbell differs significantly from the reverse complement of the other strand. Subread orientation is inferred from the alignment. Small heteroduplexes, such as single base `A` paired with a matching `G`, are retained and the ambiguity is reflected in base quality. Molecules with a single difference longer than 20 bp between the strands are removed and recorded as heteroduplexes in the `<outputPrefix>.ccs_report.txt` file.

6. Trim large insertions

- Insertions in the subreads relative to the draft that are longer than `--max-insertion-size`, default 30 bp, are trimmed since they typically represented spurious sequencing activity.

7. Filter candidates

- For each window, a heuristic is used to find those positions that likely need polishing. In addition, homopolymers are always polished. Skipping unambiguous positions makes the polishing at least twice as fast.

8. Polishing

- The core polishing uses the `arrow` algorithm, a left-right Hidden Markov-Model (HMM) that models the enzymatic and photophysical events in SMRT sequencing. Emission and transition parameters are estimated by a dinucleotide template context. Transition parameters form a homogeneous Markov chain. The transition parameters do **not** depend on the position within the template, only the pulse width of a base call, the dinucleotide context of the template, and the SNR of the ZMW. `Arrow` computes the log-likelihood that the subread originates from the template, marginalizing over all possible alignments of the subread to the template. For every position in the template that is a candidate for polishing, `arrow` tests if the log-likelihood is improved by substituting one of the other three nucleotides, inserting one of the four nucleotides after the position, or deleting the position itself. Once `arrow` does not find any further beneficial mutations to the template in an iteration, it stops.

9. QV calculation

- The log-likelihood ratio between the most likely template sequence and all of its mutated counterparts is used to calculate a quality for each base in the final consensus. The average of the per-base qualities is the read accuracy, `rq`.

10. Final steps

- The per-window consensus template sequences and base qualities are concatenated and overhangs (overlaps between adjacent windows) are trimmed. If the predicted read accuracy is at least `--min-rq`, then the consensus read is written to the output.

Input files

- One `.subreads.bam` file containing the subreads for each SMRTbell® template sequenced.

Output files

- A BAM file with one entry for each consensus sequence derived from a productive ZMW. BAM is a general file format for storing sequence data, which is described fully by the SAM/BAM working group. The CCS analysis output format is a version of this general format, where the consensus sequence is represented by the "Query Sequence". Several tags were added to provide additional meta information. An example BAM entry for a consensus as seen by `samtools` is shown below.

```
m64009_201008_223950/1/ccs      4      *      0      255      *      *      0      0      ATCGCCTACC
~|~t~R~r~      RG:Z:a773clf2      fi:B:C,26,60,21,41,33,26,63,45,73,33      fn:i:6
fp:B:C,11,18,21,35,8,18,31,8,23,11      np:i:12
ri:B:C,17,37,24,4,70,21,12,44,21,32      rn:i:6      rp:B:C,16,56,17,9,23,19,10,10,23,12
rq:f:0.999651      sn:B:f,10.999,16.2603,3.964,7.17746      we:i:9816064      ws:i:20
zm:i:1
```

Following are some of the common fields contained in the output BAM file:

Field	Description
Query Name	Movie Name / ZMW # /ccs
FLAG	Required by the format but meaningless in this context. Always set to 4 to indicate the read is unmapped.
Reference Name	Required by the format but meaningless in this context. Always set to *.
Mapping Start	Required by the format but meaningless in this context. Always set to 0.
Mapping Quality	Required by the format but meaningless in this context. Always set to 255.
CIGAR	Required by the format but meaningless in this context. Always set to *.
RNEXT	Required by the format but meaningless in this context. Always set to *.
PNEXT	Required by the format but meaningless in this context. Always set to 0.
TLEN	Required by the format but meaningless in this context. Always set to 0.
Consensus Sequence	The consensus sequence generated.
Quality Values	The per-base parametric quality metric. For details see "Interpreting QUAL values" on page 12.
RG Tag	The read group identifier.
bc Tag	A 2-entry array of upstream-provided barcode calls for this ZMW.
bq Tag	The quality of the barcode call. (Optional: Depends on barcoded inputs.)

Field	Description
np Tag	The number of full passes that went into the subread. (Optional : Depends on barcoded inputs.)
rq Tag	The predicted read quality.
zm Tag	The ZMW hole number.
ma Tag	Bitmask storing whether a SMRTbell adapter is missing on either side of the molecule that produced a CCS read. 0 indicates that neither end has a confirmed missing adapter.
ac Tag	An array containing the counts of detected and missing SMRTbell adapters on either side of the molecule that produced a CCS read: <ul style="list-style-type: none"> • Detected adapters on left/start • Missing adapters on left/start • Detected adapters on right/end • Missing adapter on right/end

Usage

```
ccs [OPTIONS] INPUT OUTPUT
```

Example

```
ccs --all myData.subreads.bam myResult.bam
```

Required	Description
Input File Name	The name of a single subreads.bam or a subreadset.xml file to be processed. (Example = myData.subreads.bam)
Output File Name	The name of the output BAM file; comes after all other options listed. Valid output files are the BAM and the Dataset.xml formats. (Example = myResult.bam)

Options	Description
--version	Prints the version number.
--report-file	Contains a result tally of the outcomes for all ZMWs that were processed. If no file name is given, the report is output to the file <code>ccs_report.txt</code> . In addition to the count of successfully-produced consensus sequences, this file lists how many ZMWs failed various data quality filters (SNR too low, not enough full passes, and so on) and is useful for diagnosing unexpected drops in yield.
--min-snr	Removes data that is likely to contain deletions. SNR is a measure of the strength of signal for all 4 channels (A, C, G, T) used to detect base pair incorporation. This value sets the threshold for minimum required SNR for any of the four channels. Data with SNR < 2.5 is typically considered lower quality. (Default = 2.5)
--min-length	Specifies the minimum length requirement for the minimum length of the draft consensus to be used for further polishing. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates. (Default = 10)
--max-length	Specifies the maximum length requirement for the maximum length of the draft consensus to be used for further polishing. For robust results while avoiding unnecessary computation on unusual data, set to ~20% above the largest expected insert size. (Default = 50000)
--min-passes	Specifies the minimum number of passes for a ZMW to be emitted. This is the number of full passes. Full passes must have an adapter hit before and after the insert sequence and so do not include any partial passes at the start and end of the sequencing reaction. It is computed as the number of passes made across all windows. (Default = 3)

Options	Description
<code>--min-rq</code>	Specifies the minimum predicted accuracy of a read. CCS analysis generates an accuracy prediction for each read, defined as the expected percentage of matches in an alignment of the consensus sequence to the true read. A value of <code>0.99</code> indicates that only reads expected to be 99% accurate are emitted. (Default = <code>0.99</code>)
<code>--num-threads</code>	Specifies how many threads to use while processing. By default, CCS analysis uses as many threads as there are available cores to minimize processing time, but fewer threads can be specified here.
<code>--log-file</code>	The name of a log file to use. If none is given, the logging information is printed to <code>STDERR</code> . (Example: <code>mylog.txt</code>)
<code>--log-level</code>	Specifies verbosity of log data to produce. By setting <code>--logLevel=DEBUG</code> , you can obtain detailed information on what ZMWs were dropped during processing, as well as any errors which may have appeared. (Default = <code>INFO</code>)
<code>--skip-polish</code>	After constructing the draft consensus, do not proceed with the polishing steps. This is significantly faster, but generates less accurate data with no RQ or QUAL values associated with each base.
<code>--by-strand</code>	Separately generates a consensus sequence from the forward and reverse strands. Useful for identifying heteroduplexes formed during sample preparation.
<code>--chunk</code>	Operates on a single chunk. Format <code>i/N</code> , where <code>i</code> in <code>[1,N]</code> . Examples: <code>3/24</code> or <code>9/9</code> .
<code>--max-chunks</code>	Determines the maximum number of chunks, given an input file.
<code>--model-path</code>	Specifies the path to a model file or directory containing model files.
<code>--model-spec</code>	Specifies the name of the chemistry or model to use, overriding the default selection.
<code>--all</code>	<p>Generates one representative sequence per ZMW, irrespective of quality and passes. <code>--min-passes 0 --min-rq 0 --max-length 0</code> are set implicitly and cannot be changed; <code>--all</code> also deactivates the maximum draft length filter. Filtering has to be performed downstream.</p> <p>The <code>ccs --all</code> option changes the workflow as follows:</p> <ol style="list-style-type: none"> 1. There is special behavior for low-pass ZMWs. If a ZMW has fewer than 2 full-length subreads, use the subread of median length as representative consensus, optionally with its kinetic information as forward orientation using <code>--all-kinetics</code>, and do not polish. 2. Only polish ZMWs with at least two full-length subreads mapping back to the draft. Otherwise, set predicted accuracy <code>rq</code> tag to <code>-1</code> to indicate that the predicted accuracy was not calculated, and populate per-base QVs with <code>+</code> (QV 10) the approximate raw accuracy. Kinetic information is not available for unpolished drafts. 3. Instead of using an unpolished draft without kinetic information as a representative consensus sequence, if <code>--subread-fallback</code> is used, fall back to a representative subread with kinetic information. <p>How is <code>--all</code> different from explicitly setting <code>--min-passes 0 --min-rq 0</code>?</p> <ul style="list-style-type: none"> • Setting <code>--min-passes 0 --min-rq 0</code> is a brute-force combination that polishes every ZMW, even those that only have one partial subread, with polishing making no difference. In contrast, <code>--all</code> is a bit smarter and only polishes ZMWs with at least one full-length subread and one additional partial subread.

Options	Description
<code>--hifi-kinetics</code>	<p>Generates averaged kinetic information for polished reads, independently for both strands of the insert. Forward is defined with respect to the orientation represented in SEQ and is considered to be the native orientation. As with other PacBio-specific tags, aligners will not re-orient these fields.</p> <p>Base modifications can be inferred from per-base pulse width (PW) and inter-pulse duration (IPD) kinetics.</p> <p>Minor cases exist where a certain orientation may get filtered out entirely from a ZMW, preventing valid values from being passed for that record. In these cases, empty lists are passed for the respective record/orientation, and number of passes are set to zero.</p> <p>To facilitate the use of HiFi reads with base modifications workflows, we added an executable in pbbam called <code>ccs-kinetics-bystrandify</code> which creates a pseudo <code>--by-strand</code> BAM with corresponding <code>pw</code> and <code>ip</code> tags that imitates a normal, unaligned subreads BAM.</p>
<code>--all-kinetics</code>	Adds kinetic information for all ZMWs, except for unpolished draft consensus.
<code>--subread-fallback</code>	When combined with <code>--all</code> , uses a subread instead of a draft as representative consensus.
<code>--suppress-reports</code>	Suppresses the generation of default reports and metric files.

Interpreting QUAL values

The QUAL value of a read is a measure of the posterior likelihood of an error at a particular position. **Increasing** QUAL values are associated with a **decreasing** probability of error. For indels and homopolymers, there is ambiguity as to which QUAL value is associated with the error probability. Shown below are different types of alignment errors, with an * indicating which sequence BP should be associated with the alignment error.

Mismatch

```

      *
ccs:  ACGTATA
ref:  ACATATA

```

Deletion

```

      *
ccs:  AC-TATA
ref:  ACATATA

```

Insertion

```

      *
ccs:  ACGTATA
ref:  AC-TATA

```

Homopolymer insertion or deletion

Indels should always be left-aligned, and the error probability is only given for the first base in a homopolymer.

```

      *                      *
ccs:  ACGGGGTATA    ccs:  AC-GGGTATA
ref:  AC-GGGTATA    ref:  ACGGGGTATA

```


CCS Analysis Yield report

The CCS Analysis Yield report specifies the number of ZMWs that successfully produced consensus sequences, as well as a count of how many ZMWs did **not** produce a consensus sequence for various reasons. The entries in this report, as well as parameters used to increase or decrease the number of ZMWs that pass various filters, are shown in the table below.

The first part is a summary of inputs and outputs:

ZMW results	Parameters affecting results	Description
ZMWs input	None	The number of input ZMWs.
ZMWs pass filters	All custom processing settings	The number of CCS reads successfully produced on the first attempt, using the fast windowed approach.
ZMWs fail filters	All custom processing settings	The number of ZMWs reads that failed to produce a CCS read.
ZMWs shortcut filters	-all	The number of ZMWs having fewer than 2 full-length subreads.
ZMWs with tandem repeats	--min-tandem-repeat-length	The number of ZMWs with repeats larger than the specified threshold.

The second part explains in detail the exclusive ZMW count for those ZMWs that were filtered:

ZMW results	Parameters affecting results	Description
No usable subreads	None	The ZMW had no usable subreads. Either there were no subreads, or all subreads had lengths outside the range <50% or >200% of the median subread length.
Below SNR threshold	--min-snr	The ZMW had at least one channel's SNR below the minimum threshold.
Lacking full passes	--min-passes	There were not enough subreads that had an adapter at the start and end of the subread (a "full pass").
Heteroduplexes	None	The SMRTbell contains a heteroduplex. In this case, it is not clear what the consensus should be and so the ZMW is dropped.
Min coverage violation	None	The ZMW is damaged on one strand and cannot be polished reliably.
Draft generation error	None	Subreads do not match the generated draft sequence, even after multiple tries.
Draft above --max-length	--max-length	The draft sequence was above the maximum length threshold.
Draft below --min-length	--min-length	The draft sequence was below the minimum length threshold.
Lacking usable subreads	None	Too many subreads were dropped while polishing.
CCS analysis did not converge	None	The consensus sequence did not converge after the maximum number of allowed rounds of polishing.

ZMW results	Parameters affecting results	Description
CCS read below minimum predicted accuracy	<code>--min-rq</code>	Each CCS read has a predicted level of accuracy associated with it. Reads that are below the minimum specified threshold are removed.
Unknown error during processing	None	These should not occur.

How do I read the `ccs_report.txt` file?

By default, each CCS analysis generates a `ccs_report.txt` file. This file summarizes how many ZMWs generated HiFi reads and how many ZMWs failed CCS reads generation because of the listed causes. For those failing, each ZMW contributes to exactly one reason of failure; percentages are with respect to number of failed ZMWs.

Does CCS analysis dislike low-complexity regions?

Low-complexity comes in many shapes and forms. A particular challenge for CCS analysis are highly-enriched tandem repeats, such as hundreds of copies of AGGGGT. Prior to `ccs` v5.0, inserts with many copies of a small repeat likely did not generate a consensus sequence. Since `ccs` v5.0, every ZMW is tested if it contains a tandem repeat of length

`--min-tandem-repeat-length 1000`. For this, we use symmetric DUST, specifically [this](#) `sdust` implementation, but slightly modified. If a ZMW is flagged as a tandem repeat, internally `--disable-heuristics` is activated for only this ZMW, and various filters that are known to exclude low-complexity sequences are disabled. This recovers most of the low-complexity consensus sequences, without impacting run time performance.

Can I produce one consensus sequence for each strand of a molecule?

Yes, use `--by-strand`. Make sure that you have sufficient coverage, as `--min-passes` are per-strand in this case. For each strand, CCS analysis generates one consensus read that has to pass all filters. Read name suffix indicates strand. **Example:**

```
m64011_190714_120746/14/ccs/rev
m64011_190714_120746/35/ccs/fwd
```

How does `--by-strand` work? For each ZMW:

- Determine orientation of reads with respect to the one closest to the median length.
- Sort reads into two buckets, forward and reverse strands.
- Treat each strand as an individual entity as we do with ZMWs:
 - Apply all filters per strand individually.
 - Create a draft for each strand.

- Polish each strand.
- Write out each polished strand consensus.

BAM tags generated

Tag	Type	Description
ec	f	Effective coverage
fi	B,C	Forward IPD (Codec V1)
fn	i	Forward number of complete passes (zero or more)
fp	B,C	Forward PulseWidth (Codec V1)
np	i	Number of full-length subreads
ri	B,C	Reverse IPD (Codec V1)
rn	i	Reverse number of complete passes (zero or more)
rp	B,C	Reverse PulseWidth (Codec V1)
rq	f	Predicted average read accuracy
sn	B,F	Signal-to-noise ratios for each nucleotide
zm	i	ZMW hole number
RG	z	Read group

dataset The `dataset` tool creates, opens, manipulates and writes Data Set XML files. The commands allow you to perform operations on the various types of data held by a Data Set XML: Merge, split, write, and so on.

Usage

```
dataset [-h] [--version] [--log-file LOG_FILE]
        [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug | --quiet | -v]
        [--strict] [--skipCounts]
```

{create,filter,merge,split,validate,summarize,consolidate,loadstats,newuuid,loadmetadata,copyto,absolutize,relativize}

Options	Description
-h, --help	Displays help information and exits.
<Command> -h	Displays help for a specific command.
-v, --version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. (Default = None, writes to stdout.)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL]. (Default = INFO)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
-v	Sets the verbosity level. (Default = NONE)
--strict	Turns on strict tests and display all errors. (Default = False)
--skipCounts	Skips updating NumRecords and TotalLength counts. (Default = False)

create Command: Create an XML file from a `fofn` (file-of-file names) or **BAM** file. Possible types: SubreadSet, AlignmentSet, ReferenceSet, HdfSubreadSet, BarcodeSet, ConsensusAlignmentSet, ConsensusReadSet, ContigSet.

```
dataset create [-h] [--type DSTYPE] [--name DSNAME] [--generateIndices]
               [--metadata METADATA] [--novalidate] [--relative]
               outfile infile [infile ...]
```

Example

The following example shows how to use the `dataset create` command to create a barcode file:

```
dataset create --generateIndices --name my_barcodes --type BarcodeSet
my_barcodes.barcodeset.xml my_barcodes.fasta
```

Required	Description
outfile	The name of the XML file to create.
infile	The <code>fofn</code> (file-of-file-names) or BAM file(s) to convert into an XML file.

Options	Description
--type DSTYPE	Specifies the type of XML file to create. (Default = NONE)
--name DSNAME	The name of the new Data Set XML file.
--generateIndices	Generates index files (.pbi and .bai for BAM, .fai for FASTA). Requires samtools/pysam and pbindex. (Default = FALSE)
--metadata METADATA	A metadata.xml file (or Data Set XML) to supply metadata. (Default = NONE)
--novalidate	Specifies not to validate the resulting XML. Leaves the paths as they are.
--relative	Makes the included paths relative instead of absolute. This is not compatible with --novalidate.

`filter` Command: Filter an XML file using filters and threshold values.

- **Suggested filters:** alignedlength, as, astart, bc, bcf, bcq, bcr, bq, cx, length, mapqv, movie, n_subreads, pos, qend, qid, qname, qstart, readstart, rname, rq, tend, tstart, zm
- **More resource-intensive filter:** [qs]

Note: Multiple filters with different names are ANDed together. Multiple filters with the **same** name are ORed together, duplicating existing requirements. The filter string should be enclosed in **single quotes**.

```
dataset filter [-h] infile outfile filters [filters ...]
```

Required	Description
infile	The name of the XML file to filter.
outfile	The name of the output filtered XML file.
filters	The values to filter on. (Example: rq>0.85)

Examples

Filter on read quality > 0.99 (Q20):

```
% dataset filter in.consensusreadset.xml hifi.consensusreadset.xml 'rq >= 0.99'
```

Filter on read quality and length:

```
% dataset filter in.consensusreadset.xml filtered.consensusreadset.xml 'rq >= 0.99 AND length >= 10000'
```

Filter for very long and very short reads:

```
% dataset filter in.consensusreadset.xml filtered.consensusreadset.xml 'length >= 40000; length <= 400'
```

Filter for specific high-quality barcodes:

```
% dataset filter mixed.consensusreadset.xml samples1-3.consensusreadset.xml 'bc = [0,1,2] AND bq >= 26'
```

merge Command: Combine XML files.

```
dataset merge [-h] outfile infiles [infiles ...]
```

Required	Description
infiles	The names of the XML files to merge.
outfile	The name of the output XML file.

split Command: Split a Data Set XML file.

```
dataset split [-h] [--contigs] [--barcodes] [--zmws] [--byRefLength]
               [--noCounts] [--chunks CHUNKS] [--maxChunks MAXCHUNKS]
               [--targetSize TARGETSIZE] [--breakContigs]
               [--subdatasets] [--outdir
infiles [outfiles...]
```

Required	Description
infile	The name of the XML file to split.

Options	Description
outfiles	The names of the resulting XML files.
--contigs	Splits the XML file based on contigs. (Default = FALSE)
--barcodes	Splits the XML file based on barcodes. (Default = FALSE)
--zmws	Splits the XML file based on ZMWs. (Default = FALSE)
--byRefLength	Splits contigs by contig length. (Default = TRUE)
--noCounts	Updates the Data Set counts after the split. (Default = FALSE)
--chunks x	Splits contigs into x total windows. (Default = 0)
--maxChunks x	Splits the contig list into at most x groups. (Default = 0)
--targetSize x	Specifies the minimum number of records per chunk. (Default = 5000)
--breakContigs	Breaks contigs to get closer to maxCounts. (Default = False)
--subdatasets	Splits the XML file based on sub-datasets. (Default = False)
--outdir OUTDIR	Specifies an output directory for the resulting XML files. (Default = <in-place>, not the current working directory.)

validate Command: Validate XML and ResourceId files. (This is an internal testing functionality that may be useful.)

Note: This command requires that `pyxb` (**not** distributed with SMRT Link) be installed. If **not** installed, `validate` simply checks that the files pointed to in `ResourceIds` exist.

```
dataset validate [-h] [--skipFiles] infile
```

Required	Description
infile	The name of the XML file to validate.

Options	Description
--skipFiles	Skips validating external resources. (Default = False)

`summarize` Command: Summarize a Data Set XML file.

```
dataset summarize [-h] infile
```

Required	Description
infile	The name of the XML file to summarize.

`consolidate` Command: Consolidate XML files.

```
dataset consolidate [-h] [--numFiles NUMFILES] [--noTmp]
infile datafile xmlfile
```

Required	Description
infile	The name of the XML file to consolidate.
datafile	The name of the resulting data file.
xmlfile	The name of the resulting XML file.

Options	Description
--numFiles x	Specifies the number of data files to produce. (Default = 1)
--noTmp	Do not copy to a temporary location to ensure local disk use. (Default = False)

`loadstats` Command: Load an `sts.xml` file containing pipeline statistics into a Data Set XML file.

```
dataset loadstats [-h] [--outfile OUTFILE] infile statsfile
```

Required	Description
infile	The name of the Data Set XML file to modify.
statsfile	The name of the <code>.sts.xml</code> file to load.

Options	Description
--outfile OUTFILE	The name of the XML file to output. (Default = None)

`newuuid` Command: Refresh a Data Set's Unique ID.

```
dataset newuuid [-h] [--random] infile
```

Required	Description
<code>infile</code>	The name of the XML file to refresh.

Options	Description
<code>--random</code>	Generates a random UUID, instead of a hash. (Default = <code>False</code>)

`loadmetadata` Command: Load a `.metadata.xml` file into a Data Set XML file.

```
dataset loadmetadata [-h] [--outfile OUTFILE] infile metadata
```

Required	Description
<code>infile</code>	The name of the Data Set XML file to modify.
<code>metadata</code>	The <code>.metadata.xml</code> file to load, or Data Set to borrow from.

Options	Description
<code>--outfile OUTFILE</code>	Specifies the XML file to output. (Default = <code>None</code>)

`copyto` Command: Copy a Data Set and resources to a new location.

```
dataset copyto [-h] [--relative] infile outdir
```

Required	Description
<code>infile</code>	The name of the XML file to copy.
<code>outdir</code>	The directory to copy to.

Options	Description
<code>--relative</code>	Makes the included paths relative instead of absolute. (Default = <code>False</code>)

`absolutize` Command: Make the paths in an XML file absolute.

```
dataset absolutize [-h] [--outdir OUTDIR] infile
```

Required	Description
<code>infile</code>	The name of the XML file whose paths should be absolute.

Options	Description
<code>--outdir OUTDIR</code>	Specifies an optional output directory. (Default = <code>None</code>)

relativize Command: Make the paths in an XML file relative.

```
dataset relativize [-h] infile
```

Required	Description
infile	The name of the XML file whose paths should be relative.

Examples - Filter reads

To filter one or more BAM file's worth of subreads, aligned or otherwise, and then place them into a single BAM file:

```
# usage: dataset filter <in_fn.xml> <out_fn.xml> <filters>
dataset filter in_fn.subreadset.xml filtered_fn.subreadset.xml 'rq>0.85'

# usage: dataset consolidate <in_fn.xml> <out_data_fn.bam> <out_fn.xml>
dataset consolidate filtered_fn.subreadset.xml consolidate.subreads.bam
out_fn.subreadset.xml
```

The filtered Data Set and the consolidated Data Set should be read-for-read equivalent when used with SMRT® Analysis software.

Example - Resequencing pipeline

- Align two movie's worth of subreads in two SubreadSets to a reference.
- Merge the subreads together.
- Split the subreads into Data Set chunks by contig.
- Process using `gcpp` on a chunkwise basis (in parallel).

1. Align each movie to the reference, producing a Data Set with one BAM file for each execution:

```
pbalgn movie1.subreadset.xml referenceset.xml movie1.alignmentset.xml
pbalgn movie2.subreadset.xml referenceset.xml movie2.alignmentset.xml
```

2. Merge the files into a FOFN-like Data Set; BAMs are not touched:

```
# dataset merge <out_fn> <in_fn> [<in_fn> <in_fn> ...]
dataset merge merged.alignmentset.xml movie1.alignmentset.xml movie2.alignmentset.xml
```

3. Split the Data Set into chunks by contig name; BAMs are not touched:
 - Note that supplying output files splits the Data Set into that many output files (up to the number of contigs), with multiple contigs per file.
 - **Not** supplying output files splits the Data Set into **one** output file per contig, named automatically.
 - Specifying a number of chunks instead will produce that many files, with contig or even subcontig (reference window) splitting.

```
dataset split --contigs --chunks 8 merged.alignmentset.xml
```

4. Process the chunks:

```
gcpp --reference referenceset.xml --output  
chunk1consensus.fasta,chunk1consensus.fastq,chunk1consensus.vcf,chunk1consensus.gff  
chunk1contigs.alignmentset.xml
```

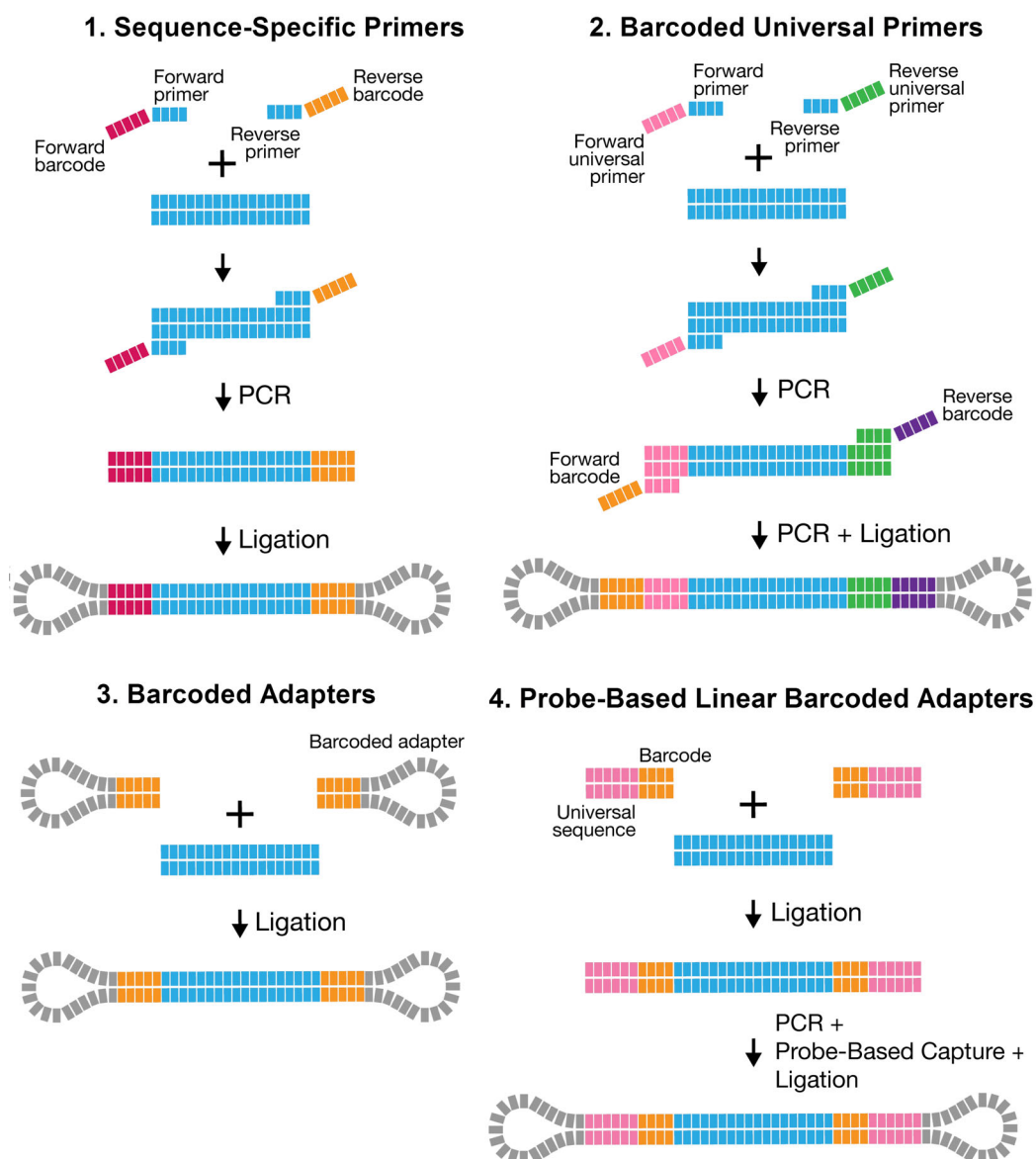
The chunking works by duplicating the original merged Data Set (no BAM duplication) and adding filters to each duplicate such that only reads belonging to the appropriate contigs are emitted. The contigs are distributed among the output files in such a way that the total number of records per chunk is about even.

Demultiplex Barcodes

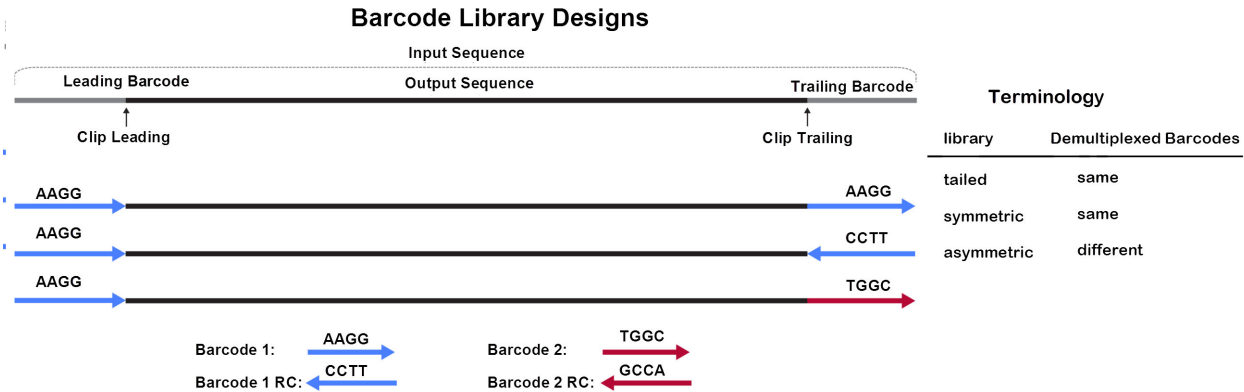
The **Demultiplex Barcodes** application identifies barcode sequences in PacBio single-molecule sequencing data.

Demultiplex Barcodes can demultiplex samples that have a unique per-sample barcode pair and were pooled and sequenced on the same SMRT® Cell. There are four different methods for barcoding samples with PacBio technology:

1. Sequence-specific primers
2. Barcoded universal primers
3. Barcoded adapters
4. Linear Barcoded Adapters for Probe-based Captures



In addition, there are three different barcode library designs. **Demultiplex Barcodes** supports raw subread and CCS reads demultiplexing.



In the overview above, the input sequence is flanked by adapters on both sides. The bases adjacent to an adapter are **barcode regions**. A read can have up to two barcode regions, leading and trailing. Either or both adapters can be missing and consequently the leading and/or trailing region is not being identified.

For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For barcode identification, one read with a single barcode region is sufficient.

For the **asymmetric** design, **different** barcodes are attached to the sides of the insert sequence of interest. To identify the different barcodes, a read with leading and trailing barcode regions is required.

Output barcode pairs are generated from the identified barcodes. The barcode names are combined using "--", for example `bc1002--bc1054`. The sort order is defined by the barcode indices, starting with the lowest.

Workflow

By default, **Demultiplex Barcodes** processes input reads grouped by ZMW, **except** if the `--per-read` option is used. All barcode regions along the read are processed individually. The final per-ZMW result is a summary over all barcode regions. Each ZMW is assigned to a pair of selected barcodes from the provided set of candidate barcodes. Subreads from the same ZMW will have the same barcode and barcode quality. For a particular target barcode region, every barcode sequence gets aligned as given and as reverse-complement, and higher scoring orientation is chosen. This results in a list of scores over all candidate barcodes.

- If only **same** barcode pairs are of interest (symmetric/tailed), use the `--same` option to filter out **different** barcode pairs.

- If only **different** barcode pairs are of interest (asymmetric), use the `--different` option to require at least two barcodes to be read, and remove pairs with the **same** barcode.

Parameter presets

Recommended parameter combinations are available using `--preset` for HiFi input:

- **HIFI-SYMMETRIC**
`--ccs --min-score 80 --min-end-score 50 --min-ref-span 0.75`
`--same`
- **HIFI-ASYMMETRIC**
`--ccs --min-score 80 --min-end-score 50 --min-ref-span 0.75`
`--different --min-scoring-regions 2`
- **NONE (Default)**

Half adapters

For an adapter call with only one barcode region, the high-quality region finder cuts right through the adapter. The preceding or succeeding subread was too short and was removed, or the sequencing reaction started/stopped there. This is called a **half adapter**. Thus, there are also 1.5, 2.5, N+0.5 adapter calls.

ZMWs with half or only one adapter can be used to identify the same barcode pairs; positive-predictive value might be reduced compared to high adapter calls. For asymmetric designs with different barcodes in a pair, at least a single full-pass read is required; this can be two adapters, two half adapters, or a combination.

Usage:

- Any existing output files are **overwritten** after execution.
- Always use `--peek-guess` to remove spurious barcode hits.

Analysis of subread data:

```
lima movie.subreads.bam barcodes.fasta prefix.bam
lima movie.subreadset.xml barcodes.barcodeset.xml prefix.subreadset.xml
```

Analysis of CCS reads:

```
lima --css movie.ccs.bam barcodes.fasta prefix.bam
lima --ccs movie.consensusreadset.xml barcodes.barcodeset.xml
prefix.consensusreadset.xml
```

If you do not need to import the demultiplexed data into SMRT Link, use the `--no-pbi` option to minimize memory consumption and run time.

Symmetric or tailed options:

```
Raw: --same
CCS read: --same --ccs
```

Asymmetric options:

Raw: --different

CCS reads: --different --ccs

Example execution:

```
lima m54317_180718_075644.subreadset.xml \
Sequel_RSII_384_barcode_v1.barcode.v1 \
m54317_180718_075644.demux.subreadset.xml \
--different --peek-guess
```

Options	Description
--same	Retains only reads with the same barcodes on both ends of the insert sequence, such as symmetric and tailed designs.
--different	Retains only reads with different barcodes on both ends of the insert sequence, asymmetric designs. Enforces --min-passes ≥ 1.
--min-length n	Omits reads with lengths below n base pairs after demultiplexing. ZMWs with no reads passing are omitted. (Default = 50)
--max-input-length n	Omits reads with lengths above n base pairs for scoring in the demultiplexing step. (Default = 0, deactivated)
--min-score n	Omits ZMWs with average barcode scores below n. A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator how well the chosen barcode pair matches. It is normalized to a range between 0 (no hit) and 100 (a perfect match). (Default = 0, PacBio recommends setting it to 26.)
--min-end-score n	Specifies the minimum end barcode score threshold applied to the individual leading and trailing ends. (Default = 0)
--min-passes n	Omits ZMWs with less than n full passes, a read with a leading and trailing adapter. (Default = 0, no full-pass needed) Example: 0 pass : insert - adapter - insert 1 pass : insert - adapter - INSERT - adapter - insert 2 passes: insert - adapter - INSERT - adapter - INSERT - adapter - insert
--score-full-pass	Uses only reads flanked by adapters on both sides (full-pass reads) for barcode identification.
--min-ref-span	Specifies the minimum reference span relative to the barcode length. (Default = 0.5)
--per-read	Scores and tags per subread, instead of per ZMW.
--ccs	Sets defaults to -A 1 -B 4 -D 3 -I 3 -X 1.
--peek n	Looks at the first n ZMWs of the input and return the mean. This lets you test multiple test barcode.fasta files and see which set of barcodes was used.
--guess n	This performs demultiplexing twice. In the first iteration, all barcodes are tested per ZMW. Afterwards, the barcode occurrences are counted and their mean is tested against the threshold n; only those barcode pairs that pass this threshold are used in the second iteration to produce the final demultiplexed output. A prefix.lima.guess file shows the decision process; --same is being respected.
--guess-min-count	Specifies the minimum ZMW count to whitelist a barcode. This filter is ANDed with the minimum barcode score specified by --guess. (Default = 0)

Options	Description
<code>--peek-guess</code>	Sets the following options: <code>--peek 50000 --guess 45 --guess-min-count 10</code> . Demultiplex Barcodes will run twice on the input data. For the first 50,000 ZMWs, it will guess the barcodes and store the mask of identified barcodes. In the second run, the barcode mask is used to demultiplex all ZMWs. If combined with <code>--ccs</code> then the barcode score threshold is increased by <code>--guess 75</code> .
<code>--single-side</code>	Identifies barcodes in molecules that only have barcodes adjacent to one adapter.
<code>--window-size-mult</code> <code>--window-size-bp</code>	The candidate region size multiplier: <code>barcode_length * multiplier</code> . (Default = 3) Optionally, you can specify the region size in base pairs using <code>--window-size-bp</code> . If set, <code>--window-size-mult</code> is ignored.
<code>--num-threads n</code>	Spawns <code>n</code> threads; 0 means use all available cores. This option also controls the number of threads used for BAM and PBI compression. (Default = 0)
<code>--chunk-size n</code>	Specifies that each thread consumes <code>n</code> ZMWs per chunk for processing. (Default = 10).
<code>--no-bam</code>	Does not produce BAM output. Useful if only reports are of interest, as run time is shorter.
<code>--no-pbi</code>	Does not produce a <code>.bam.pbi</code> index file. The on-the-fly <code>.bam.pbi</code> file generation buffers the output data. If you do not need a <code>.bam.pbi</code> index file for SMRT Link import, use this option to decrease memory usage to a minimum and shorten the run time.
<code>--no-reports</code>	Does not produce any reports. Useful if only demultiplexed BAM files are needed.
<code>--dump-clips</code>	Outputs all clipped barcode regions generated to the <code><prefix>.lima.clips</code> file.
<code>--dump-removed</code>	Outputs all records that did not pass the specified thresholds, or are without barcodes, to the <code><prefix>.lima.removed.bam</code> file.
<code>--split-bam</code> <code>--split-bam-named</code>	Specifies that each barcode has its own BAM file called <code>prefix.idxBest-idxCombined.bam</code> , such as <code>prefix.0-0.bam</code> . Optionally, <code>--split-bam-named</code> names the files by their barcode names instead of their barcode indices.
<code>--isoseq</code>	Removes primers as part of the Iso-Seq® pipeline. See “Demultiplexing Iso-Seq® data” on page 32 for details.
<code>--bad-adapter-ratio n</code>	Specifies the maximum ratio of bad adapters. (Default = 0).

Input files

Input data in PacBio-enhanced BAM format is either:

- Sequence data - Unaligned subreads, directly from Sequel II systems and Sequel IIe systems.
- Unaligned CCS reads, generated by CCS analysis.

Barcodes are provided as a FASTA file or BarcodeSet file:

- One entry per barcode sequence.
- **No** duplicate sequences.

- All bases must be in **upper-case**.
- Orientation-agnostic (forward or reverse-complement, but **not** reversed.)

Example:

```
>bc1000
CTCTACTTACTTACTG
>bc1001
GTCGTATCATCATGTA
>bc1002
AATATACCTATCATTA
```

Note: Name barcodes using an alphabetic character prefix to avoid later barcode name/index confusion.

Output files

Demultiplex Barcodes generates multiple output files by default, all starting with the same prefix as the output file, using the suffixes `.bam`, `.subreadset.xml`, and `.consensusreadset.xml`. The report prefix is `lima`. **Example:**

```
lima m54007_170702_064558.subreads.bam barcode.fasta /my/path/
m54007_170702_064558_demux.subreadset.xml
```

For all output files, the prefix is

`/my/path/m54007_170702_064558_demux.`

- `<prefix>.bam`: Contains clipped records, annotated with barcode tags, that passed filters and respect the `--same` option.
- `<prefix>.lima.report`: A tab-separated file describing each ZMW, unfiltered. This is useful information for investigating the demultiplexing process and the underlying data. A single row contains **all** reads from a single ZMW. For `--per-read`, each row contains one subread, and ZMWs might span multiple rows.
- `<prefix>.lima.summary`: Lists how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.

(1)

```
ZMWs input (A): 213120
ZMWs above all thresholds (B): 176356 (83%)
ZMWs below any threshold (C): 36764 (17%)
```

(2)

```
ZMW Marginals for (C):
Below min length : 26 (0%)
Below min score : 0 (0%)
Below min end score : 5138 (13%)
Below min passes : 0 (0%)
Below min score lead : 11656 (32%)
Below min ref span : 3124 (8%)
Without adapter : 25094 (68%)
With bad adapter : 10349 (28%) <- Only with --bad-adapter-ratio
```



```

Undesired hybrids           : xxx (xx%) <- Only with --peek-guess
Undesired same barcode pairs : xxx (xx%) <- Only with --different
Undesired diff barcode pairs : xxx (xx%) <- Only with --same
Undesired 5p--5p pairs      : xxx (xx%) <- Only with --isoseq
Undesired 3p--3p pairs      : xxx (xx%) <- Only with --isoseq
Undesired single side       : xxx (xx%) <- Only with --isoseq
Undesired no hit            : xxx (xx%) <- Only with --isoseq

```

(3)

```

ZMWs for (B):
With same barcode           : 162244 (92%)
With different barcodes     : 14112 (8%)
Coefficient of correlation   : 32.79%

```

(4)

```

ZMWs for (A):
Allow diff barcode pair     : 157264 (74%)
Allow same barcode pair     : 188026 (88%)
Bad adapter yield loss      : 10112 (5%) <- Only with --bad-adapter-ratio
Bad adapter impurity        : 10348 (5%) <- Only without --bad-adapter-ratio

```

(5)

```

Reads for (B):
Above length                 : 1278461 (100%)
Below length                 : 2787 (0%)

```

Explanation of each block:

1. Number of ZMWs that went into `lima`, how many ZMWs were passed to the output file, and how many did not qualify.
2. For those ZMWs that did not qualify: The marginal counts of each filter. (Filter are described in the Options table.)
When running with `--peek-guess` or similar manual option combination and different barcode pairs are found during peek, the full SMRT Cell may contain low-abundant different barcode pairs that were identified during peek individually, but not as a pair. Those unwanted barcode pairs are called hybrids.
3. For those ZMWs that passed: How many were flagged as having the same or different barcode pair, as well as the coefficient of variation for the barcode ZMW yield distribution in percent.
4. For all input ZMWs: How many allow calling the same or different barcode pair. This is a simplified version of how many ZMW have at least one full pass to allow a different barcode pair call and how many ZMWs have at least half an adapter, allowing the same barcode pair call.
5. For those ZMWs that qualified: The number of reads that are above and below the specified `--min-length` threshold.
 - `<prefix>.lima.counts: A .tsv` file listing the counts of each observed barcode pair. Only passing ZMWs are counted.

Example: `column -t prefix.lima.count`

IdxFirst	IdxCombined	IdxFirstNamed	IdxCombinedNamed	Counts	MeanScore
0	0	bc1001	bc1001	1145	68
1	1	bc1002	bc1002	974	69
2	2	bc1003	bc1003	1087	68

- `<prefix>.lima.clips`: Contains clipped barcode regions generated using the `--dump-clips` option. **Example:**

```
head -n 6 prefix.lima.clips
>m54007_170702_064558/4850602/6488_6512 bq:34 bc:11
CATGTCCCTCAGTTAAGTTACAA
>m54007_170702_064558/4850602/6582_6605 bq:37 bc:11
TTTGTACTAAGTATACCAATAG
>m54007_170702_064558/4916040/4801_4816 bq:93 bc:10
```
- `<prefix>.lima.removed.bam`: Contains records that did **not** pass the specified thresholds, or are without barcodes, using the option `--dump-removed.lima` does **not** generate a `.pbi`, nor Data Set for this file. This option **cannot** be used with any splitting option.
- `<prefix>.lima.guess`: A `.tsv` file that describes the barcode subsetting process activated using the `--peek` and `--guess` options.

IdxFirst	IdxCombined	IdxFirstNamed	IdxCombinedNamed	NumZMWs	MeanScore	Picked
0	0	bc1001t	bc1001t	1008	50	1
1	1	bc1002t	bc1002t	1005	60	1
2	2	bc1003t	bc1003t	5	24	0
3	3	bc1004t	bc1004t	555	61	1

- One `DataSet`, `.subreadset.xml`, or `.consensusreadset.xml` file is generated per output BAM file.
- `.pbi`: One PBI file is generated per output BAM file.

What is a universal spacer sequence and how does it affect demultiplexing?

For library designs that include an identical sequence between adapter and barcode, such as probe-based linear barcoded adapters samples, Demultiplex Barcodes offers a special mode that is activated if it finds a shared prefix sequence among all provided barcode sequences.

Example:

```
>custombc1
ACATGACTGTGACTATCTCACACATATCAGAGTGCG
>custombc2
ACATGACTGTGACTATCTCAACACACAGACTGTGAG
```

In this case, Demultiplex Barcodes detects the shared prefix

ACATGACTGTGACTATCTCA and removes it internally from all barcodes. Subsequently, it increases the window size by the length L of the prefix sequence.

- If `--window-size-bp N` is used, the actual window size is $L + N$.
- If `--window-size-mult M` is used, the actual window size is $(L + |bc|) * M$.

Because the alignment is semi-global, a leading reference gap can be added without any penalty to the barcode score.

What are bad adapters?

In the `subreads.bam` file, each subread has a context flag `cx`. The flag specifies, among other things, whether a subread has flanking adapters, before and/or after. Adapter-finding was improved and can also find molecularly-missing adapters, or those obscured by a local decrease in accuracy. This may lead to missing or obscured bases in the flanking barcode. Such adapters are labelled "bad", as they don't align with the adapter reference sequence(s). Regions flanking those bad adapters are problematic, because they can fully or partially miss the barcode bases, leading to wrong classification of the molecule. `lima` can handle those adapters by **ignoring** regions flanking bad adapters. For this, `lima` computes the ratio of number of bad adapters divided by number of all adapters.

By default, `--bad-adapter-ratio` is set to 0 and does **not** perform any filtering. In this mode, bad adapters are handled just like good adapters.

But the `*.lima.summary` file contains one row with the number of ZMWs that have at least 25% bad adapters, but otherwise pass all other filters. This metric can be used as a diagnostic to assess library preparation.

If `--bad-adapter-ratio` is set to non-zero positive $(0, 1)$, bad adapter flanking barcode regions are treated as missing. If a ZMW has a higher ratio of bad adapters than provided, the ZMW is filtered and consequently removed from the output. The `*.lima.summary` file contains two additional rows.

```
With bad adapter      : 10349 (28%)
Bad adapter yield loss : 10112 (5%)
```

The first row counts the number of ZMWs that have bad adapter ratios that are too high; the percentage is with respect to the number of all ZMW not passing. The second row counts the number of ZMWs that are removed solely due to bad adapter ratios that are too high; the percentage is with respect the number of all input ZMWs and consequently is the effective yield loss caused by bad adapters.

If a ZMW has ~50% bad adapters, one side of the molecule is molecularly-missing an adapter. For 100% bad adapter, **both** sides are missing adapters. A lower than ~40% percentage indicates decreased local accuracy during sequencing leading to adapter sequences not being found. If a high percentage of ZMWs is molecularly-missing adapters, you should improve library preparation.

Demultiplexing Iso-Seq® data

Demultiplex Barcodes is used to identify and remove Iso-Seq cDNA primers. If the Iso-Seq sample is barcoded, the barcodes should be included as part of the primer. Only by using the command-line can users use `lima` with the `--isoseq` option for demultiplexing Iso-Seq data.

The input Iso-Seq data format for demultiplexing is `.ccs.bam`. Users must first generate a CCS reads BAM file for an Iso-Seq Data Set before running `lima`. The recommended parameters for running CCS analysis for Iso-Seq are `min-pass=1,min accuracy=0.9`, and turning Polish to OFF.

1. Primer IDs must be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should not include the Ts and is written in reverse complement.
2. Below are four example primer sets. The first is unbarcoded, the second has barcodes (shown in lower case) adjacent to the 3' primer.

Example 1: The Iso-Seq cDNA Primer primer set, included with the SMRT Link installation.

Users following the standard Iso-Seq Express protocol **without** multiplexing, or running a Data Set that has **already** been demultiplexed (either using Run Design or the SMRT® Analysis application) should use this default option.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>IsoSeq_3p
GTACTCTGCGTTGATACCACTGCTT
```

Example 2: The Iso-Seq 12 Barcoded cDNA Primers set, included with the SMRT Link installation.

Users using barcoded cDNA primers listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document

Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell Prep Kit 3.0, should select this option.

```
>bc1001_5p
CACATATCAGAGTGC GGCAATGAAGTCGCAGGGTTGGGG
>bc1002_5p
ACACACAGACTGTGAGGCAATGAAGTCGCAGGGTTGGGG
...
```

(There are a total of 24 sequence records, representing 12 pairs of F/R barcoded cDNA primers.)

Example 3: An example of a custom cDNA primer set. 4 tissues were multiplexed using barcodes on the 3' end only.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>dT_BC1001_3p
AAGCAGTGGTATCAACGCAGAGTACCACATATCAGAGTGC
>dT_BC1002_3p
AAGCAGTGGTATCAACGCAGAGTACACACACAGACTGTGAG
>dT_BC1003_3p
AAGCAGTGGTATCAACGCAGAGTACACACATCTCGTGAGAG
>dT_BC1004_3p
AAGCAGTGGTATCAACGCAGAGTACCACGCACACACGCGCG
```

Example 4: Special Handling for the TeloPrime cDNA Kit

The Lexogen TeloPrime cDNA kit contains **As** in the 3' primer that **cannot** be differentiated from the polyA tail. For best results, remove the **As** from the 3' end as shown below:

```
>TeloPrimeModified_5p
TGGATTGATATGTAATACGACTCACTATAG
>TeloPrimeModified_3p
CGCCTGAGA
```

3. Use the `--isoseq` mode. Note that this **cannot** be combined with the `--guess` option.

4. The output will be only different pairs with a 5p and 3p combination:

```
demux.5p--tissue1_3p.bam
demux.5p--tissue2_3p.bam
```

The `--isoseq` parameter set is very conservative for removing any spurious and ambiguous calls, and guarantees that only proper asymmetric (barcoded) primer are used in downstream analyses. Good libraries reach >75% CCS reads passing the Demultiplex Barcodes filters.

BAM tags

In SMRT Link v11.0, **LB** and **SM** tags are set by the user in Run Design. The **SM** tag can also be set in Demultiplex Barcodes in SMRT Analysis.

Non-demultiplex case:

- LB: Well Sample Name.
- SM: Bio Sample Name.

Multiplexed case, BAM pre-demultiplexing:

- LB: Well Sample Name.
- SM: Tag removed.

Multiplexed case, BAMs post-demultiplexing:

- LB: Well Sample Name for all child barcode BAMs.
- SM: Each individual Bio Sample Name for the specific barcode.
- BC: Barcode sequence or hyphenated barcode sequences of the pair.
- DS: Appends barcode information used in demultiplexing: BarcodeFile, BarcodeHash, BarcodeCount, BarcodeMode, BarcodeQuality.
- Example read group header after demultiplexing:

```
@RG
ID:66d5a6af/3--3
PL:PACBIO
DS:READTYPE=SUBREAD;
  Ipd:CodecV1=ip;
  PulseWidth:CodecV1=pw;
  BINDINGKIT=101-500-400;
  SEQUENCINGKIT=101-427-800;
  BASECALLERVERSION=5.0.0;
  FRAMERATEHZ=100.000000;
  BarcodeFile=Sequel_16_barcode_v3.barcode.set.xml;
  BarcodeHash=f2b1fa0b43eb6ccbb30749883bb550e3;
  BarcodeCount=16;
  BarcodeMode=Symmetric;
  BarcodeQuality=Score
PU:m54010_200212_162236
SM:MySampleName
PM:SEQUEL
BC:ACAGTCGAGCGCTGCGT
```

export-datasets The `export-datasets` tool takes one or more PacBio Data Set XML files and packages all contents (including index files and supplemental Data Sets) into a single ZIP archive. Data Set resources, such as BAM files, are reorganized and renamed to flatten the directory structure, avoid redundant file writes, and convert all resource paths from absolute paths to relative paths. Where multiple Data Sets are provided, the contents of each is nested in a directory named after the `UniqueId` attribute in the XML.

The resulting archive is primarily intended to be directly imported into SMRT Link using the Data Management interface, but it may also be unpacked manually and used on the command line.

Usage

```
export-datasets [options] <dataset>...
```

Options	Description
<code>-o, --output</code>	Name of output ZIP file. (Default = <code>datasets_<timestamp>.zip</code>)
<code>--keep-parent-ref</code>	Keeps the reference to the parent Data Set when archiving a demultiplexed child Data Set.
<code>--no-scrap</code>	Excludes the <code>scrap.bam</code> file if present in the XML file.
<code>-h, --help</code>	Displays help information and exits.
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are <code>[ERROR, DEBUG, INFO, WARN]</code> . (Default = <code>WARN</code>)
<code>--logback</code>	Override all logger configuration using a specified <code>logback.xml</code> file.
<code>--log2stdout</code>	If <code>True</code> , log output is displayed to the console. (Default = <code>False</code>)
<code>--debug</code>	Alias for setting the log level to <code>DEBUG</code> . (Default = <code>False</code>)
<code>--quiet</code>	Alias for setting the log level to <code>ERROR</code> . (Default = <code>False</code>)
<code>--verbose</code>	Alias for setting the log level to <code>INFO</code> . (Default = <code>False</code>)

Input files

- One or more PacBio Dataset XML files.

Output file

- One output ZIP file.

Examples

```
export-datasets m64001_200704_012345.subreadset.xml
```

```
export-datasets sample1.consensusreadset.xml sample2.consensusreadset.xml
\sample3.consensusreadset.xml -o barcoded_ccs.zip
```

```
export-datasets /opt/smrtlink/jobs/0000/0000001/0000001234/outputs/
mapped.alignmentset.xml
```

export-job The `export-job` tool packages a SMRT Link Analysis job for export to another system, usually for reimportation into another SMRT Link instance. All internal paths in job output files are converted from absolute to relative paths, and many of the internal details of the Cromwell workflows are omitted. The export is **not** a complete record of the job, but rather a collection of job output files and metadata.

Note that `export-job` **will** include any external Data Sets referenced in output Data Sets inside the job, for example ReferenceSets associated with mapped Data Sets, or BarcodeSets associated with demultiplexed Data Sets. However, these Data Sets will **not** be imported along with the job. The exported job does **not** include the input reads used to run the job; these may be exported separately using the `export-datasets` tool.

Important: Only SMRT Link v10.0 or later generates the necessary metadata files for `export-job` to save a full record of job execution. Jobs created with older versions of SMRT Link will still be archived, but the metadata will be empty and/or incorrect.

Usage

```
export-job [options] <job_dir>
```

Options	Description
<job_dir>	Path to a SMRT Link job directory.
-o, --output	Name of output ZIP file. (Default = job_<timestamp>.zip)
-h, --help	Displays help information and exits.
--log-file	Writes the log to a file. (Default = stderr)
--log-level	Specifies the log level; values are [ERROR, DEBUG, INFO, WARN]. (Default = WARN)
--logback	Override all logger configuration using a specified logback.xml file.
--log2stdout	If True, log output is displayed to the console. (Default = False)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to ERROR. (Default = False)
--verbose	Alias for setting the log level to INFO. (Default = False)

Input

- A path to a job directory.

Output file

- One output ZIP file.

Examples

```
export-job /path/to/smrtlink/jobs-root/0000/0000000/0000000860 -o job860.zip
```


To reimport on another system:

```
pbservice import-job job860.zip
```

gcpp `gcpp` is a variant-calling tool provided by the `GCpp` package which provides several variant-calling algorithms for PacBio sequencing data.

Usage

```
gcpp      -j8 --algorithm=arrow \
          -r lambdaNEB.fa        \
          -o variants.gff        \
          aligned_subreads.bam
```

This example requests variant-calling, using 8 worker processes and the Arrow algorithm, taking input from the file `aligned_subreads.bam`, using the FASTA file `lambdaNEB.fa` as the reference, and writing output to `variants.gff`.

A particularly useful option is `--referenceWindow/-w`; which allows the variant-calling to be performed exclusively on a **window** of the reference genome.

Input files

- A sorted file of reference-aligned reads in PacBio's standard BAM format.
- A FASTA file that follows the PacBio FASTA file convention. If specifying an input FASTA file, a FASTA index file (`.fai`) with the same name and path is **required**. If the `.fai` file is not supplied, `gcpp` exits and displays an error message.

Note: The `--algorithm=arrow` option requires that certain metrics be in place in the input BAM file. It requires per-read SNR metrics, and the per-base `PulseWidth` metric for Sequel data.

The selected algorithm will stop with an error message if any features that it requires are unavailable.

Output files

Output files are specified as comma-separated arguments to the `-o` flag. The file name extension provided to the `-o` flag is meaningful, as it determines the output file format. For example:

```
gcpp aligned_subreads.bam -r lambda.fa -o myVariants.gff,myConsensus.fasta
```

will read input from `aligned_subreads.bam`, using the reference `lambda.fa`, and send variant call output to the file `myVariants.gff`, and consensus output to `myConsensus.fasta`.

The file formats currently supported (using extensions) are:

- `.gff`: PacBio GFFv3 variants format; convertible to BED.
- `.vcf`: VCF 4.2 variants format (that is compatible with v4.3.)
- `.fasta`: FASTA file recording the consensus sequence calculated for each reference contig.

- `.fastq`: FASTQ file recording the consensus sequence calculated for each reference contig, as well as per-base confidence scores.

Options	Description
<code>-j</code>	Specifies the number of worker processes to use.
<code>--algorithm=</code>	Specifies the variant-calling algorithm to use; values are <code>plurality</code> , <code>arrow</code> and <code>poa</code> . (Default = <code>arrow</code>)
<code>-r</code>	Specifies the FASTA reference file to use.
<code>-o</code>	Specifies the output file format; values are <code>.gff</code> , <code>.vcf</code> , <code>.fasta</code> , and <code>.fastq</code> .
<code>--maskRadius</code>	When using the <code>arrow</code> algorithm, setting this option to a value <code>N</code> greater than 0 causes <code>gcpp</code> to pass over the data a second time after masking out regions of reads that have >70% errors in $2*N+1$ bases. This setting has little to no effect at low coverage, but for high-coverage datasets (>50X), setting this parameter to 3 may improve final consensus accuracy. In rare circumstances, such as misassembly or mapping to the wrong reference, enabling this parameter may cause worse performance.
<code>--minConfidence MINCONFIDENCE</code> <code>-q MINCONFIDENCE</code>	Specifies the minimum confidence for a variant call to be output to variants.{gff,vcf} (Default = 40)
<code>--minCoverage MINCOVERAGE</code> <code>-x MINCOVERAGE</code>	Specifies the minimum site coverage for variant calls and consensus to be calculated for a site. (Default = 5)

Available algorithms

At this time there are three algorithms available for variant calling: `plurality`, `poa` and `arrow`.

- `plurality` is a simple and very fast procedure that merely tallies the most frequent read base or bases found in alignment with each reference base, and reports deviations from the reference as potential variants. This approach is prone to insertion and deletion errors.
- `poa` uses the partial order alignment algorithm to determine the consensus sequence. It is a heuristic algorithm that approximates a multiple sequence alignment by progressively aligning sequences to an existing set of alignments.
- `arrow` uses the per-read SNR metric and the per-pulse `pulsewidth` metric as part of its likelihood model.

Confidence values

The `arrow` and `plurality` algorithms make a confidence metric available for every position of the consensus sequence. The confidence should be interpreted as a phred-transformed posterior probability that the consensus call is incorrect; such as:

$$QV = -10\log_{10}(p_{err})$$

`gcpp` clips reported QV values at 93; larger values **cannot** be encoded in a standard FASTQ file.

Chemistry specificity

The `--algorithm=arrow` parameter is trained per-chemistry. `arrow` identifies the sequencing chemistry used for each run by looking at metadata contained in the input BAM data file. This behavior can be overridden by a command-line option.

When multiple chemistries are represented in the reads in the input file, the Arrow will model reads appropriately using the parameter set for its chemistry, thus yielding optimal results.

Genome Assembly

The Genome Assembly application generates *de novo* assemblies using HiFi reads. The application is fast, produces contiguous assemblies, and is suitable for genomes of any size.

The Genome Assembly application is powered by the IPA HiFi genome assembler and includes the following features:

- Separates haplotypes during assembly using a novel phasing stage (Nighthawk).
- Polishes the contigs with phased reads using `Racon`.
- Improves haplotype separation using the `purge_dups` tool.

Workflow of the Genome Assembly application

Analysis steps are highly optimized to produce assemblies of large genomes efficiently.



The workflow consists of seven stages:

1. Sequence database construction.
2. Fast overlap computation using the `Pancake` tool.
3. A dedicated phasing stage using the `Nighthawk` tool.
4. Filtering chimeras and residual repeats.
5. Layout based on the string graph.
6. Polishing using the `Racon` tool.
7. Purging haplotype duplicates from the primary assembly using the third-party tool `purge_dups`.

The workflow accepts HiFi XML Data Sets as input.

IPA HiFi genome assembler

- Scales well on a cluster.
- The workflow has an embedded downsampling feature:
 - If the genome size and the desired coverage are specified, the initial stage (sequence database construction) downsamples the input Data Set to the desired coverage.
 - Otherwise, the full coverage is used.

Usage

The Genome Assembly application is run using the `pbccromwell run` command, with the `pb_assembly_hifi` parameter to specify the application. See [“pbccromwell” on page 75](#) for details.

To view information on the available Genome Assembly options, enter:

```
pbcrumwell show-workflow-details pb_assembly_hifi
```

The **minimum** command needed to run the workflow requires the input and the number of threads. The following example uses 16 threads:

```
pbcrumwell run pb_assembly_hifi -e <input.xml> --nproc 16
```

The following example performs an assembly using an input XML Data Set, and uses all default settings, including 1 CPU:

```
pbcrumwell run pb_assembly_hifi -e <input.consensusreadset.xml>
```

Note: The default options for this workflow are equivalent to the following command:

```
pbcrumwell run pb_assembly_hifi \  
-e <input.consensusreadset.xml> \  
--task-option reads=None \  
--task-option ipa2_genome_size=0 \  
--task-option ipa2_downsampled_coverage=0 \  
--task-option ipa2_advanced_options="" \  
--task-option ipa2_run_polishing=True \  
--task-option ipa2_run_phasing=True \  
--task-option ipa2_run_purge_dups=True \  
--task-option ipa2_ctg_prefix="ctg." \  
--task-option ipa2_reads_db_prefix="reads" \  
--task-option ipa2_cleanup_intermediate_files=True \  
--task-option dataset_filters="" \  
--task-option filter_min_qv=20 \  
--nproc 8
```

The default options for this workflow should work well for any genome types.

If the assembly is run on a single local node with high CPU count, such as 64 cores, we recommend that the job submission for `pbcrumwell` is configured so that it uses 4 concurrent jobs and 16 threads per job.

We found this to be more efficient than using 64 threads and 1 concurrent job, as many steps are very data I/O-dependent.

You can apply a similar principle for compute environments with more or fewer cores. For example, for a machine with 80 cores, one can use 20 threads and 4 concurrent jobs.

Genome Assembly parameters input files

Option	Default value	Description
<code>-e, --eid_ccs</code>	NONE	Optional parameter, required if <code>--task-option reads</code> <code><input></code> is not specified. This is a SMRT Link-specific input parameter and supports only PacBio Consensusreadset XML files as input.
<code>--task-option reads</code>	NONE	Optional parameter, required if <code>-e <input></code> is not specified. Supports multiple input formats: FASTA, FASTQ, BAM, XML, FOFN and gzipped versions of FASTA/FASTQ.
<code>--task-option ipa2_genome_size</code>	0	The approximate number of base pairs expected in the genome. This is used only for downsampling; if the value is ≤ 0 , downsampling is disabled. Note: It is better to slightly overestimate rather than underestimate the genome length to ensure good coverage across the genome.
<code>--task-option ipa2_downsampled_coverage</code>	0	The input Data Set can be downsampled to a desired coverage, provided that both the <code>ipa2_downsampled_coverage</code> and <code>ipa2_genome_size</code> options are specified and >0 . Downsampling applies to the entire assembly process, including polishing. This parameter selects reads randomly, using a fixed random seed for reproducibility.
<code>--task-option ipa2_advanced_options</code>	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.)
<code>--task-option ipa2_run_polishing</code>	TRUE	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies.
<code>--task-option ipa2_run_phasing</code>	TRUE	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies.
<code>--task-option ipa2_run_purge_dups</code>	TRUE	Enables or disables identification of "duplicate" alternate haplotype contigs which may be assembled in the primary contig file, and moves them to the associate contig (haplotig) file.
<code>--task-option ipa2_ctg_prefix</code>	.ctg	The prefix used to label the output generated contigs.
<code>--task-option ipa2_reads_db_prefix</code>	reads	The prefix of the sequence and seed databases which will be used internally for assembly.
<code>--task-option ipa2_cleanup_intermediate_files</code>	TRUE	Removes intermediate files from the run directory to save space.
<code>--task-option dataset_filters</code>	NONE	(General pbcromwell option) A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
<code>--task-option filter_min_qv</code>	20	(General pbcromwell option) Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
<code>--task-option downsample_factor</code>	0	Downsampling factor applied directly to the input Data Set. This parameter is not related to <code>ipa2_downsampled_coverage</code> .
<code>--task-option mem_scale_factor</code>	8	Controls the amount of requested memory for individual Cromwell tasks in the workflow. The default value of 8 is good for larger genomes, but it may be too much memory for smaller genomes.

Option	Default value	Description
<code>--config</code>	NONE	(General pb cromwell option) Java configuration file for running Cromwell.
<code>--nproc</code>	1	(General pb cromwell option) Number of processors, (except per task).

- *.bam file containing PacBio data.
- *.fasta or *.fastq file containing PacBio data.
- *.xml file containing PacBio data.
- *.fofn files with file names of files containing PacBio data.

Output files

- `final_purged_primary.fasta` file containing assembled primary contigs.
- `final_purged_haplotigs.fasta` file containing assembled haplotigs.

Advanced parameters

Advanced parameters should be rarely modified. For the special cases when that is required, advanced parameters are documented below.

Advanced parameters specified on the command line:

- Are in the form of `key = value` pairs.
- Each pair is separated by a semicolon (;) character.
- The full set of advanced parameters is surrounded by **one** set of double quotes.
- The specified value of a parameter **overwrites** the default options for that key. **All** desired options of that parameter must be explicitly listed, not just the ones which should change from the default.
- Setting an empty value **clears** the parameter; it does **not** reset the value back to default.

Example:

```
--task-option ipa2_advanced_options="config_seeddb_opt=-k 28;config_block_size=2048"
```


Complete list of available advanced parameters and default values:

Advanced parameters	Default value	Description
config_genome_size	0	The approximate number of base pairs expected in the genome, used to determine the coverage cutoff. This is only used for downsampling; 0 turns downsampling off. Note: It is better to slightly overestimate rather than underestimate the genome length to ensure good coverage across the genome.
config_coverage	0	The input Data Set can be downsampled to a desired coverage, provided that both the <code>Downsampled coverage</code> and <code>Genome Length</code> parameters are specified and above 0. Downsampling applies to the entire assembly process, including polishing. This feature selects reads randomly, using a fixed random seed for reproducibility.
config_polish_run	1	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies. 0 disables this feature; 1 enables it.
config_phase_run	1	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies. 0 disables this feature; 1 enables it.
config_purge_dups_run	1	Enables or disables the <code>purge_dups</code> stage of the workflow. 0 disables this feature; 1 enables it.
config_autocomp_max_cov	1	If enabled, the maximum allowed overlap coverage at either the 5' or the 3' end of every read is automatically determined based on the statistics computed from the overlap piles. This value is appended to the <code>config_ovl_filter_opt</code> value internally, and supersedes the manually specified <code>--max-cov</code> and <code>--max-diff</code> values of that parameter. These options are used to determine potential repeats and filter out those reads before the string graph is constructed. 0 disables this feature; 1 enables it.
config_block_size	4096	The overlapping process is performed on pairs of blocks of input sequences, where each block contains the number of sequences which crop up to this size (in Mbp). Note: The number of pairwise comparisons grows quadratically with the number of blocks (meaning more cluster jobs), but also the larger the block size the more resources are required to execute each pairwise comparison.
config_existing_db_prefix	NONE	Allows injection of an existing SeqDB, so that one doesn't have to be built from scratch. The provided existing DB is symbolically linked and used for assembly. (This option is intended for debugging purposes.)

Advanced parameters	Default value	Description
config_ovl_filter_opt	--max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 4000 --gapFilt --minDepth 4 --idt-stage2 98	<p>Overlap filter options.</p> <p>--gapFilt - Enables the chimera filter, which analyzes each overlap pile, and determines whether a pread is chimeric based on the local coverage across the pread.</p> <p>--minDepth - Option for the chimera filter. The chimera filter is ignored when a local region of a read has coverage lower than this value.</p> <p>The other parameters are:</p> <p>--min-cov - Minimum allowed coverage at either the 5' or the 3' end of a read. If the coverage is below this value, the read is blacklisted and all of the overlaps it is incident with are ignored. This helps remove potentially chimeric reads.</p> <p>--max-cov - Maximum allowed coverage at either the 5' or the 3' end of a read. If the coverage is above this value, the read is blacklisted and all of the overlaps it is incident with are ignored. This helps remove repetitive reads which can make tangles in the string graph. Note that this value is a heuristic which works well for ~30x seed length cutoff. If the cutoff is set higher, we advise that this value be also increased. Alternatively, using the <code>autocompute_max_cov</code> option can automatically estimate the value of this parameter, which can improve contiguity (for example, in cases when the input genome size or the seed coverage were overestimated).</p> <p>--max-diff - Maximum allowed difference between the coverages at the 5' and 3' ends of any particular read. If the coverage is above this value, the read is blacklisted and all of the overlaps it is incident with are ignored. If the <code>autocompute_max_cov</code> option is used, then the same computed value is supplied to this parameter as well.</p> <p>--bestn - Keep at most this many overlaps on the 5' and the 3' side of any particular read.</p> <p>--min-len - Filter overlaps where either A-read or the B-read are shorter than this value.</p> <p>--idt-stage2 - Filter overlaps with identity below 98%.</p> <p>--high-copy-sample-rate - Controls the downsampling of reads from high copy elements to the expected coverage determined by <code>maxCov*rate</code>, where <code>rate</code> is the value of this parameter. If <code>rate</code> is 0, then these high coverage reads are discarded.</p>
config_ovl_min_idt	98	The final overlap identity threshold. Applied during the final filtering stage, right before the overlaps are passed to the layout stage.
config_ovl_min_len	1000	The minimum length of either A-read or a B-read to keep the overlap. Applied during the final filtering stage, right before the overlaps are passed to the layout stage.
config_ovl_opt	--one-hit-per-target --min-idt 96	<p>Overlapping options for the <code>pancake</code> overlapping tool. The options set by this parameter here are passed directly to <code>pancake</code>. For details on <code>pancake</code> options, use <code>pancake -h</code>.</p> <p>The defaults used here are: <code>--one-hit-per-target</code> which keeps only the best hit in case there are multiple possible overlaps between a pair of reads (tandem repeats); and <code>--min-idt 96</code> which will filter out any overlap with identity lower than 96%.</p>
config_phasing_opt	NONE	Options for the phasing tool <code>nighthawk</code> . The options set by this parameter are passed directly to <code>nighthawk</code> . For details on <code>nighthawk</code> options, use <code>nighthawk -h</code> .

Advanced parameters	Default value	Description
config_phasing_split_opt	--split-type noverlaps --limit 3000000	Options that control the chunking of the phasing jobs, and through that regulate the time and memory consumption of each individual chunk. The defaults are: --split-type overlaps which splits the chunks by the number of overlaps; and --limit 3000000 which allow at most approximately 3 million overlaps per chunk. Empirically, the current defaults keep the maximum memory consumption (RSS) of the phasing jobs under 4 GB per chunk.
config_seeddb_opt	-k 28 -w 120 --space 1	Options to control the seed computation. These options are passed directly to the <code>pancake seeddb</code> command. Defaults: -k 28 is the k-mer size of 28 bp; -w 120 is the minimizer window size of 120 bp; and --space 1 specifies the spacing for spaced seed construction, with 1 gap in between every two bases of the seed. For more details on these and other options, use <code>pancake seeddb -h</code> .
config_seqdb_opt	--compression 1	Options to control the construction of the sequence database. These options are passed directly to the <code>pancake seqdb</code> command. Current default is --compression 1 which turns on the 2-bit encoding compression of the sequences. For more details on these and other options, use <code>pancake seqdb -h</code> .
config_use_hpc	0	This parameter enables (1) or disables (0) an experimental Homopolymer Compression feature. If this feature is enabled, the overlaps are computed from homopolymer-compressed sequences. The layout stage is somewhat slower because the sequences have to be aligned to determine the correct homopolymer-expanded coordinates.
config_use_seq_ids	1	This feature is mostly useful for debugging purposes. If 0 is specified, then the overlaps contain original sequence names instead of their numerical IDs. The default of 1 uses the numerical IDs to represent reads, which uses memory much more efficiently.
config_purge_map_opt	--min-map-len 1000 --min-idt 98.0 --bestn 5	This option is used to control the mapping of the reads to contigs for the <code>purge_dups</code> tool. The mapper used is <code>pancake</code> , and the options set by this parameter are used directly by <code>pancake</code> . For details on <code>pancake</code> options, use <code>pancake -h</code> . Option --min-map-len 1000 removes any alignment which did not span more than 1000 bp during the mapping process; --min-idt 98.0 removes any alignment with identity below 98.0%, and --bestn 5 keeps at most 5 top scoring alignments for each query read (one primary alignment and at most 4 secondary alignments).

Advanced parameters	Default value	Description
<code>config_purge_dups_calcuts</code>	NONE	<p>The third-party tool <code>purge_dups</code> can accept user-defined cutoffs for purging. On some genomes, the automated computation of the cutoffs in <code>purge_dups</code> can result in suboptimal values, and in this case a user can specify them manually.</p> <p>This option is passed directly to the <code>purge_dups_calcuts</code> tool. For details on the possible values that can be passed to this tool, use <code>ipa_purge_dups_calcuts</code> without parameters.</p> <p>Relevant parameters include:</p> <ul style="list-style-type: none"> -l INT Lower bound for read depth. -m INT Transition between haploid and diploid. -u INT Upper bound for read depth.
<code>config_m4filt_high_copy_sample_rate</code>	1.0	<p>This option is passed to the <code>--high-copy-sample-rate</code> parameter of the overlap filter, which controls the downsampling of reads from high copy elements to the expected coverage determined by <code>maxCov*rate</code>, where <code>rate</code> is the value of this parameter. If 0, then these high coverage reads are discarded.</p> <p>Note: This parameter supersedes the <code>config_ovl_filter_opt</code> options.</p>
<code>config_max_polish_block_mb</code>	100	<p>During the polishing stage, contigs are grouped into chunks of approximate size specified by this parameter (in megabases). Each chunk is processed separately and in parallel (depending on the system configuration).</p>
<code>config_layout_opt</code>	NONE	<p>This value is passed directly to the assembly layout stage. To get a list of valid options for this parameter, enter <code>ipa2_ovlp_to_graph -h</code> on the command line.</p>

HiFiViral SARS-CoV-2 Analysis

Use this application to analyze multiplexed samples sequenced with the HiFiViral SARS-CoV-2 Kit. For **each** sample, this analysis provides:

- Consensus sequence (FASTA).
- Variant calls (VCF).
- HiFi reads aligned to the reference (BAM).
- Plot of HiFi read coverage depth across the SARS-CoV-2 genome.

Across **all** samples, this analysis provides:

- Job summary table including passing sample count at 90 and 95% genome coverage.
- Sample summary table including, for each sample: Count of variable sites, genome coverage, read coverage, and probability of multiple strains, and other metrics.
- Plate QC graphical summary of performance across samples in assay plate layout.
- Plot of HiFi read depth of coverage for all samples.

Notes:

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Phred-scaled Q20.
- This application is for SARS-CoV-2 analysis **only** and is **not** recommended for other viral studies. The Wuhan reference genome is included with SMRT Link and used by default, but advanced users may specify other reference genomes. We have **not** tested the application with reference genomes other than the Wuhan reference genome.
- The application is intended to identify variable sites and call a single consensus sequence per sample. The output consensus sequence is produced based on the dominant variant observed. Minor variant information that passes through a default threshold may be encoded in the raw VCF, but does **not** get propagated into the consensus sequence FASTA.
- The HiFiViral SARS-CoV-2 Analysis application can be run using the **Auto Analysis** feature available in Run Design. This feature allows users to complete all necessary analysis steps immediately after sequencing **without** manual intervention. The Auto Analysis workflow includes CCS, Demultiplex Barcodes, and HiFiViral SARS-CoV-2 Analysis.

HiFiViral SARS-CoV-2 application workflow

1. Process the reads using the `mimux` tool to trim the probe arm sequences.
2. Align the reads to the reference genome using `pbbmm2`.
3. Call and filter variants using `bcftools`, generating the raw variant calls in VCF file format. Filtering in this step removes low-quality calls (less than Q20), and normalizes indels.

4. Filter low-frequency variants using `vcfcons` and generate a consensus sequence by injecting variants into the reference genome. At each position, a variant is called only if both the base coverage exceeds the minimum base coverage threshold (Default = 4) and the fraction of reads that support this variant is above the minimum variant frequency threshold (Default = 0.5). See [here](#) for details.

Preparing input data for the HiFiViral SARS-CoV-2 Analysis application

1. Run the Demultiplex Barcodes `cromwell` workflow, where the input to that application are HiFi reads, and the primers are multiplexed barcode primers. See “[Demultiplex Barcodes](#)” on page 23 for details. If HiFi reads have **not** been generated on the instrument, run CCS analysis first. See “[ccs](#)” on page 7 for details.
 - Provide the proper barcode sequences:
Barcoded M13 Primer Plate.
 - Use the task option `lima_symmetric_barcodes=false`. (The barcode pairs are **asymmetric**.)
- Provide the correctly-formatted barcode pair-to-Bio Sample CSV file. For details, see the `--task-option sample_wells_csv` option in the table further down this page.

Input files

- `movie.consensusreadset.xml`: Previously-demultiplexed HiFi reads, packaged as separate BAM files wrapped in an XML Data Set. (See “[Preparing input data for the HiFiViral SARS-CoV-2 Analysis application](#)” on page 50 for details.)
- `sars_cov2.referenceset.xml`: The Wuhan reference genome.
- `HiFiViral_SARS-CoV-2_Enrichment_Probes.barcodeset.xml`: Dataset XML specifying the probe sequence file in FASTA format.
- **[Optional]** `Plate_QC_csv`: Four-column CSV file that includes `barcode`, `biosample`, `plateID` and `wellID`.
To specify this optional file, add the following option to `pbserve`:
`--task-option sample_wells_csv=<path to CSV file>`

Output files

- `pb_sars_cov2_kit.probe_counts_zip`: Zipped TSV files with probe counts, per sample.
- `pb_sars_cov2_kit.variants_csv`: CSV variant calls for **all** samples.
- `pb_sars_cov2_kit.vcf_zip`: Zipped VCF files containing the final variant calls, per sample.
- `pb_sars_cov2_kit.raw_vcf_zip`: Zipped VCF files containing the raw variant calls, per sample.
- `pb_sars_cov2_kit.fasta_zip`: Zipped final consensus sequences, by sample, in FASTA format. This is a single consensus sequence with `Ns` for each sample.
- `pb_sars_cov2_kit.frag_fasta_zip`: Zipped file of consensus sequences, split on `Ns`, in FASTA format, by sample.

- `pb_sars_cov2_kit.mapped_zip`: Zipped BAM files containing the output from mapping HiFi reads to the reference genome, by sample.
- `samples.consensus_mapped.bam.zip`: Zipped BAM files containing the output from mapping consensus FASTA files to the reference genome, by sample.
- `samples.coverage.png.zip`: Zipped per-sample coverage graphs in png format.
- `hifi_reads.fastq.zip`: Zipped file of per-sample trimmed HiFi reads in FASTQ format.
- `sample_summary.csv`: Sample Summary file in CSV format, with one row per sample.

Options	Description
<code>--task-option sample_wells_csv</code>	Specifies a 4-column CSV file used to generate the Plate QC Report, which displays analysis results for each sample in the assay plate. The CSV file must contain barcode pairs, Bio Sample name, Plate IDs, and Well IDs. The report is useful for diagnosing sample issues based on plate location. (Default = None)
<code>--task-option min_coverage</code>	Specifies the minimum read coverage. Below this value, the consensus sequence will be set to Ns and no variants are called. (Default = 4)
<code>--task-option min_alt_freq</code>	Specifies that only variants whose frequency is greater than this value are reported. This frequency is determined based on the read depth (DP) and allele read count (AD) information in the VCF output file. We recommend using the default value to properly call the dominant alternative variant while also filtering out potential artifacts. (Default = 0.5)
<code>--task-option min_bq</code>	Specifies that reads with barcode scores below this minimum value are not included in analysis. (Default = 80)
<code>--task-option mimux_overrides</code>	Specifies additional options to pass to the <code>mimux</code> preprocessing tool for trimming and filtering reads by probe sequences. Options should be entered in space-separated format. Available options include: <code>--max-len</code> : Specifies the maximum sequence length. (Default = 800) <code>--same</code> : Specifies that only reads with arms sequences from the same probe are used.
<code>--task-option probes_fasta</code>	Specifies a FASTA file containing probes sequences if using probes other than those supplied with the HiFiViral SARS-CoV-2 kit.

Running the SARS-CoV-2 Analysis application

```

pbcromwell run pb_sars_cov2_kit \

-e <movie.consensusreadset.xml> \

-e $SMRT_ROOT/current/bundles/smrtinub/current/private/pacbio/barcodes/HiFiViral_SARS-
CoV-2_Enrichment_Probes.barcodeset.xml \

-e eid_ref_dataset_2:$SMRT_ROOT/current/bundles/smrtinub/current/private/pacbio/
canneddata/referenceset/SARS-CoV-2/sars_cov2.referenceset.xml

--task-option min_alt_freq=0.5 \
--task-option min_bq=80 \
--task-option mimux_overrides="--max-len=800 --same" \
--task-option sample_wells_csv=None \
--config cromwell.conf \
--nproc 8

```

ipdSummary The `ipdSummary` tool detects DNA base-modifications from kinetic signatures. It is part of the `kineticsTool` package.

`kineticsTool` loads IPDs observed at each position in the genome, compares those IPDs to value expected for unmodified DNA, and outputs the result of this statistical test. The expected IPD value for unmodified DNA can come from either an in-silico control or an amplified control. The in-silico control is trained by PacBio and shipped with the package. It predicts the IPD using the local sequence context around the current position. An amplified control Data Set is generated by sequencing unmodified DNA with the same sequence as the test sample. An amplified control sample is usually generated by whole-genome amplification of the original sample.

Modification detection

The basic mode of `kineticsTool` does an independent comparison of IPDs at each position on the genome, for each strand, and outputs various statistics to CSV and GFF files (after applying a significance filter).

Modifications identification

`kineticsTool` also has a Modification Identification mode that can decode multi-site IPD “fingerprints” into a reduced set of calls of specific modifications. This feature has the following benefits:

- Different modifications occurring on the same base can be distinguished; for example, 6mA and 4mC.
- The signal from one modification is combined into one statistic, improving sensitivity, removing extra peaks, and correctly centering the call.

Algorithm: Synthetic control

Studies of the relationship between IPD and sequence context reveal that most of the variation in mean IPD across a genome can be predicted from a 12-base sequence context surrounding the active site of the DNA polymerase. The bounds of the relevant context window correspond to the window of DNA in contact with the polymerase, as seen in DNA/polymerase crystal structures. To simplify the process of finding DNA modifications with PacBio data, the tool includes a pre-trained lookup table mapping 12-mer DNA sequences to mean IPDs observed in C2 chemistry.

Algorithm: Filtering and trimming

`kineticsTool` uses the Mapping QV generated by `pbbmm2` and stored in the `cmp.h5` or BAM file (or AlignmentSet) to **ignore** reads that are not confidently mapped. The default minimum Mapping QV required is 10, implying that `pbbmm2` has 90% confidence that the read is correctly mapped. Because of the range of read lengths inherent in PacBio data, this can be changed using the `--mapQvThreshold` option.

There are a few features of PacBio data that require special attention to achieve good modification detection performance. `kineticsTool` inspects the alignment between the observed bases and the reference sequence for an IPD measurement to be included in the analysis. The PacBio read sequence **must** match the reference sequence for k around the cognate base. In the current module, $k=1$. The IPD distribution at some locus can be thought of as a mixture between the “normal” incorporation process IPD, which is sensitive to the local sequence context and DNA modifications, and a contaminating “pause” process IPD, which has a much longer duration (mean > 10 times longer than normal), but happen rarely (~1% of IPDs).

Note: Our current understanding is that pauses do **not** carry useful information about the methylation state of the DNA; however a more careful analysis may be warranted. Also note that modifications that drastically increase the roughly 1% of observed IPDs are generated by pause events. Capping observed IPDs at the global 99th percentile is motivated by theory from robust hypothesis testing. Some sequence contexts may have naturally longer IPDs; to avoid capping too much data at those contexts, the cap threshold is adjusted per context as follows:

```
capThreshold = max(global99, 5*modelPrediction,
percentile(ipdObservations, 75))
```

Algorithm: Statistical testing

We test the hypothesis that IPDs observed at a particular locus in the sample have longer means than IPDs observed at the same locus in unmodified DNA. If we have generated a Whole Genome Amplified Data Set, which removes DNA modifications, we use a case-control, two-sample t-test. This tool also provides a pre-calibrated “synthetic control” model which predicts the unmodified IPD, given a 12-base sequence context. In the synthetic control case we use a one-sample t-test, with an adjustment to account for error in the synthetic control model.

Usage

To run using a BAM input, and output GFF and HDF5 files:

```
ipdSummary aligned.bam --reference ref.fasta m6A,m4C --gff basemods.gff \
--csv_h5 kinetics.h5
```

To run using `cmp.h5` input, perform methyl fraction calculation, and output GFF and CSV files:

```
ipdSummary aligned.cmp.h5 --reference ref.fasta m6A,m4C --methylFraction \
--gff basemods.gff --csv kinetics.csv
```

Output options	Description
<code>--gff FILENAME</code>	GFF format.
<code>--csv FILENAME</code>	Comma-separated value format.
<code>--bigwig FILENAME</code>	BigWig file format.

Input files

- A standard PacBio alignment file - either AlignmentSet XML, BAM, or `cmp.h5` - containing alignments and IPD information.
- Reference sequence used to perform alignments. This can be either a FASTA file or a ReferenceSet XML.

Output files

The tool provides results in a variety of formats suitable for in-depth statistical analysis, quick reference, and consumption by visualization tools. Results are generally indexed by reference position and reference strand. In all cases the strand value refers to the strand carrying the modification in the DNA sample. Remember that the kinetic effect of the modification is observed in read sequences aligning to the opposite strand. So reads aligning to the positive strand carry information about modification on the negative strand and vice versa, but the strand containing the putative modification is always reported.

- `modifications.gff`: Compliant with the GFF Version 3 [specification](#). Each template position/strand pair whose probability value exceeds the probability value threshold appears as a row. The template position is 1-based, per the GFF specifications. The strand column refers to the strand carrying the detected modification, which is the opposite strand from those used to detect the modification. The GFF confidence column is a Phred-transformed probability value of detection.

The auxiliary data column of the GFF file contains other statistics which may be useful for downstream analysis or filtering. These include the coverage level of the reads used to make the call, and +/- 20 bp sequence context surrounding the site.

- `modifications.csv`: Contains one row for each (reference position, strand) pair that appeared in the Data Set with coverage at least `x`. `x` defaults to 3, but is configurable with the `--minCoverage` option. The reference position index is 1-based for compatibility with the GFF file in the R environment. Note that this output type scales poorly and is **not** recommended for large genomes; the HDF5 output should perform much better in these cases.

Output columns: In-silico control mode

Column	Description
refId	Reference sequence ID of this observation.
tpl	1-based template position.
strand	Native sample strand where kinetics were generated. 0 is the strand of the original FASTA, 1 is opposite strand from FASTA.
base	The cognate base at this position in the reference.
score	Phred-transformed probability value that a kinetic deviation exists at this position.
tMean	Capped mean of normalized IPDs observed at this position.
tErr	Capped standard error of normalized IPDs observed at this position (standard deviation/sqrt(coverage)).
modelPrediction	Normalized mean IPD predicted by the synthetic control model for this sequence context.
ipdRatio	tMean/modelPrediction.
coverage	Count of valid IPDs at this position.
frac	Estimate of the fraction of molecules that carry the modification.
fracLow	2.5% confidence bound of the frac estimate.
fracUpp	97.5% confidence bound of the frac estimate.

Output columns: Case control mode

Column	Description
refId	Reference sequence ID of this observation.
tpl	1-based template position.
strand	Native sample strand where kinetics were generated. 0 is the strand of the original FASTA, 1 is opposite strand from FASTA.
base	The cognate base at this position in the reference.
score	Phred-transformed probability value that a kinetic deviation exists at this position.
caseMean	Mean of normalized case IPDs observed at this position.
controlMean	Mean of normalized control IPDs observed at this position.
caseStd	Standard deviation of case IPDs observed at this position.
controlStd	Standard deviation of control IPDs observed at this position.
ipdRatio	tMean/modelPrediction.
testStatistic	T-test statistic.
coverage	Mean of case and control coverage.
controlCoverage	Count of valid control IPDs at this position.
caseCoverage	Count of valid case IPDs at this position.

isoseq3 The `isoseq3` tool enables analysis and functional characterization of transcript isoforms for sequencing data generated on PacBio instruments. The analysis can be performed *de novo*, without a reference genome. If a reference genome is available, an optional collapse step to produce unique isoform files based on genomic coordinates is available. The input to `isoseq3` tools should be HiFi (CCS) reads in BAM format.

Usage

```
isoseq3 <tool>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits

Typical workflow

1. Visualize primers, then remove primers and demultiplex:

```
cat primers.fasta
>5p
GCAATGAAGTCGCAGGGTTGGGG
>3p
GTACTCTGCGTTGATACCACTGCTT

lima movie.ccs.bam primers.fasta demux.bam --isoseq
```

See [“Demultiplex Barcodes” on page 23](#) for details on the `lima` tool.

2. Identify and remove polyA tails; also remove artificial concatemers. The output are full-length, non-concatemer (FLNC) reads:

```
isoseq3 refine demux.5p--3p.bam primers.fasta flnc.bam --require-polya
```

3. Cluster FLNC reads at the isoform levels to generate consensus transcript isoform sequences. This generates `unpolished.hq.bam` and `unpolished.hq.fasta.gz` files, which are the high-quality (HQ) transcripts that should be analyzed further.

```
isoseq3 cluster flnc.bam unpolished.bam --use-qvs
```

4. (Optional) Map transcripts to the genome and collapse HQ transcripts based on genomic mapping:

```
pbmm2 align unpolished.bam reference.fasta aligned.sorted.bam --preset ISOSEQ --sort
isoseq3 collapse aligned.sorted.bam out.gff or
isoseq3 collapse aligned.sorted.bam movie.ccs.bam out.gff
```

See [“pbmm2” on page 84](#) for details.

refine Tool: Remove polyA and concatemers from full-length (FL) reads and generate full-length non-concatemer (FLNC) transcripts (FL to FLNC).

Usage

```
isoseq refine [options] <ccs.demux.bam|xml> <primer.fasta|xml> <flnc.bam|xml>
```

Inputs/outputs	Description
ccs.demux.bam xml	Input demultiplexed CCS reads BAM or ConsensusReadSet XML file. This is usually the output from running <code>lima</code> with the <code>--isoseq</code> option, such as <code>demux.5p--3p.bam</code> .
primer.fasta xml	Input primer FASTA or BarcodeSet XML file.
flnc.bam xml	Output FLNC BAM or ConsensusReadSet XML file.

Preprocessing	Description
<code>--min-polya-length</code>	Specifies the minimum poly(A) tail length. (Default = 20)
<code>--require-polya</code>	Requires reads to have a poly(A) tail and remove it.

Options	Description
<code>--help, -h</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--verbose, -v</code>	Sets the verbosity level.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are <code>[DEBUG, INFO, WARN, TRACE, FATAL]</code> . (Default = <code>WARN</code>)

cluster Tool: Cluster FLNC reads and generate transcripts.

Usage

```
isoseq3 cluster [options] input output
```

Example

```
isoseq3 cluster flnc.bam unpolished.bam --use-gvs
```

Custom BAM tags

`isoseq3 cluster` adds the following custom PacBio tags to the output BAM file:

- **ib:** Barcode summary: triplets delimited by semicolons, each triplet contains two barcode indices and the ZMW counts, delimited by commas. **Example:** 0,1,20;0,3,5
- **im:** ZMW names associated with this isoform.

- `is`: Number of ZMWs associated with this isoform.

Inputs/outputs	Description
input	flnc.bam file or movie.consensusreadset.xml file.
output	unpolished.bam file prefix (the actual output files will be unpolished.hq.bam and unpolished.lq-bam) or unpolished.transcriptset.xml file.

Options	Description
--s1	Specifies the number of seeds for minimizer-only clustering. (Default = 1000)
--s2	Specifies the number of seeds for DP clustering. (Default = 1000)
--poa-cov	Specifies the maximum number of CCS reads used for POA consensus. (Default = 10)
--use-qvs	Use CCS analysis Quality Values; sets --poa-cov to 100.
--split-bam	Splits BAM output files into a maximum of N files; 0 means no splitting. (Default = 0)
--min-subreads-split	Subread threshold for High-Quality/Low-Quality split; only works with --use-qvs. (Default = 7)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
-v, --verbose	Uses verbose output.
-j, --num-threads	Specifies the number of threads to use; 0 means autodetection. (Default = 0)
--log-file	Writes the log to a file. (Default = stdout)

`summarize` Tool: Create a .csv-format barcode overview from transcripts.

Usage

```
isoseq3 summarize [options] input output
```

Example

```
isoseq3 summarize unpolished.bam summary.csv
```

Inputs/outputs	Description
input	unpolished.bam file. (The output from isoseq3 cluster.)
output	summary.csv file.

Options	Description
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
-v, --verbose	Uses verbose output.
--log-file	Writes the log to file. (Default = stdout)

collapse Tool: Collapse transcripts based on genomic mapping.

Usage

```
isoseq3 collapse [options] <alignments.bam|xml> <ccs.bam|xml> <out.fastq>
```

Examples:

```
isoseq3 collapse aligned.sorted.bam out.gff
or
isoseq3 collapse aligned.sorted.bam ccs.bam out.gff
```

Inputs/outputs	Description
alignments	Alignments mapping transcripts to the reference genome. (BAM or XML file).
ccs.bam	Optional input BAM file containing CCS reads.
out.fastq	Collapsed transcripts in FASTQ format.

Options	Description
--min-aln-coverage	Ignores alignments with less than the Minimum Query Coverage. (Default = 0.95)
--min-aln-identity	Ignores alignments with less than the Minimum Alignment Identity. (Default = 0.50)
--max-fuzzy-junction	Ignores mismatches or indels shorter than or equal to N. (Default = 5)
--version	Displays program version number and exits.
--log-file	Writes the log to file. (Default = stderr)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
-j, --num-threads	Specifies the number of threads to use; 0 means autodetection. (Default = 0)

juliet `juliet` is a general-purpose minor variant caller that identifies and phases minor single nucleotide substitution variants in complex populations. It identifies codon-wise variants in coding regions, performs a reference-guided *de novo* variant discovery, and annotates known drug-resistance mutations. Insertion and deletion variants are currently ignored; support will be added in a future version. There is no technical limitation with respect to the target organism or gene.

The underlying model is a statistical test, the Bonferroni-corrected Fisher's Exact test. It compares the number of observed mutated codons to the number of expected mutations at a given position.

`juliet` uses JSON target configuration files to define different genes in longer reference sequences, such as overlapping open reading frames in HIV. These predefined configurations ease batch applications and allow immediate reproducibility. A target configuration may contain multiple coding regions within one reference sequence and optional drug resistance mutation positions.

Notes:

- The preinstalled target configurations are meant for a quick start. It is the user's responsibility to ensure that the target configurations used are correct and up-to-date.
- If the target configuration `none` was specified, the provided reference is assumed to be in-frame.

Performance

At a coverage of 6,000 CCS reads with a predicted accuracy (RQ) of ≥ 0.99 , the false positive and false negative rates are below 1% and 0.001% (10^{-5}), respectively.

Usage

```
juliet --config "HIV" data.align.bam patientZero.html
```

Required	Description
<code>input_file.bam</code>	Input aligned BAM file containing CCS reads, which must be PacBio-compliant, that is, <code>cigar M</code> is forbidden.
<code>output_file.html</code>	Output report HTML file.

Configuration	Description
<code>--config, -c</code>	Path to the target configuration JSON file, predefined target configuration tag, or the JSON string.
<code>--mode-phasing, -p</code>	Phase variants and cluster haplotypes.

Restrictions	Description
<code>--region, -r</code>	Specifies the genomic region of interest; reads are clipped to that region. Empty means all reads.
<code>--drm-only, -k</code>	Only reports DRM positions specified in the target configuration. Can be used to filter for drug-resistance mutations - only known variants from the target configuration are called.
<code>--min-perc, -m</code>	Specifies the minimum variant percentage to report. Example: <code>--min-perc 1</code> will only show variant calls with an observed abundance of more than 1%. (Default = 0)
<code>--max-perc, -n</code>	Specifies the maximum variant percentage to report. Example: <code>--max-perc 95</code> will only show variant calls with an observed abundance of less than 95%. (Default = 100)

Chemistry override (specify both)	Description
<code>--sub, -s</code>	Specifies the substitution rate. Use to override the learned rate. (Default = 0)
<code>--del, -d</code>	Specifies the deletion rate. Use to override the learned rate. (Default = 0)

Options	Description
<code>--help, -h</code>	Displays help information and exits.
<code>--verbose, -v</code>	Sets the verbosity level.
<code>--version</code>	Displays program version number and exits.
<code>--debug</code>	Returns all amino acids, irrespective of their relevance.
<code>--mode-phasing, -p</code>	Phases variants and cluster haplotypes.

Input files

- BAM-format files containing CCS reads. These must be PacBio-compliant, that is, `cigar M` is forbidden.
- Input CCS reads should have a minimal predicted accuracy of 0.99.
- Reads should be created with CCS analysis using the `--richQVs` option. Without the `--richQVs` information, the number of false positive calls might be higher, as `juliet` is missing information to filter actual heteroduplexes in the sample provided.
- `juliet` currently does **not** demultiplex barcoded data; you must provide one BAM file per barcode.

Output files

A JSON and/or HTML file:

```
juliet data.align.bam patientZero.html
juliet data.align.bam patientZero.json
juliet data.align.bam patientZero.html patientZero.json
```

The HTML file includes the same content as the JSON file, but in more human-readable format. The HTML file contains four sections:

1. Input data

Summarizes the data provided, the exact call for `juliet`, and `juliet` version for traceability purposes.

2. Target config

Summarizes details of the provided target configuration for traceability. This includes the configuration version, reference name and length, and annotated genes. Each gene name (in bold) is followed by the reference start, end positions, and possibly known drug resistance mutations.

▼ Target config

Config Version: Predefined v1.1, PacBio internal

Reference Name: HIV_HXB2

Reference Length: 9719

Genes:

- **5'LTR** (1-634)
- **p17** (790-1186)
- **p24** (1186-1879)
- **p2** (1879-1921)
- **p7** (1921-2086)
- **p1** (2086-2134)
- **p6** (2134-2292)
- **Protease** (2253-2550)
 - ATV/r: V32I L33F M46I M46L I47V G48V G48M I50L I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88S L90M
 - DRV/r: V32I L33F I47V I47A I50V I54L I54M L76V V8F I84V
 - FPV/r: V32I L33F M46I M46L I47V I47A I50V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V L90M
 - IDV/r: V32I M46I M46L I47V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V N88S L90M
 - NFV: D30N L33F M46I M46L I47V G48V G48M I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88D N88S L90M
 - SQV/r: G48V G48M I54V I54T I54A I54L I54M V82A V82T I84V N88S L90M
 - TPV/r: V32I L33F M46I M46L I47V I47A I54V I54A I54M V82T V82L I84V

3. Variant discovery

For each gene/open reading frame, there is one overview table.

Each row represents a variant position.

- Each variant position consists of the reference codon, reference amino acid, relative amino acid position in the gene, mutated codon, percentage, mutated amino acid, coverage, and possible affected drugs.
- Clicking the row displays counts of the multiple-sequence alignment counts of the -3 to +3 context positions.

▼ Variant Discovery

HIV HXB2			Reverse Transcriptase				
Codon	AA	Pos	Sample Variants				
AA	Codon	%	Coverage	Affected Drugs *			
A T G	M	41	L	T T G	1	2793	ABC + DDI + TDF + D4T + ZDV
A A A	K	65	R	A G A	1.1	2529	3TC + FTC + ABC + DDI + TDF + D4T
Pos	A	C	G	T	-	N	
-3	2947	0	0	0	0	51	
-2	2923	0	2	0	0	73	
-1	4	0	2952	0	0	42	
0	2606	0	0	0	339	53	
1	2905	0	29	0	0	64	
2	2938	0	0	0	0	60	
3	2938	0	0	0	0	60	
4	2942	0	0	0	0	56	
5	2751	0	0	0	0	247	
T A T	Y	181	C	T G T	0.91	2946	NVP + EFV + ETR + RPV
G G A	G	190	A	G C A	1	2947	NVP + EFV + ETR + RPV
A C C	T	215	Y	T A C	0.93	2877	ABC + DDI + TDF + D4T + ZDV

*HIVdb version 8.3 (last updated 2017-03-02)

► Legend

4. Drug summaries

Summarizes the variants grouped by annotated drug mutations:

▼ Drug Summaries

Drug	Gene	Reference AA	Pos	Sample AA	Sample %
3TC	Reverse Transcriptase	K	65	R	1
ABC	Reverse Transcriptase	M	41	L	0.99
		K	65	R	1
		T	215	Y	0.88

Predefined target configuration

juliet ships with one predefined target configuration, for HIV. Following is the command syntax for running that predefined target configuration:

```
juliet --config "HIV" data.align.bam patientZero.html
```

p6								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
A A C	N	47	S	A G T	0.95	2924		
Protease								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
C G A	R	8	X	T G A	0.98	2931		
Reverse Transcriptase								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
A T G	M	41	L	T T G	0.99	2903	ABC + DDI + TDF + D4T + ZDV	
A A A	K	65	R	A G A	1	2577	3TC + FTC + ABC + DDI + TDF + D4T	
T T A	L	100	F	T T T	0.85	2819		
T A T	Y	181	C	T G T	0.95	2939	NVP + EFV + ETR + RPV	
G G A	G	190	A	G C A	1	2941	NVP + EFV + ETR + RPV	
A C C	T	215	Y	T A C	0.88	2940	ABC + DDI + TDF + D4T + ZDV	

- **Note:** For the predefined configuration `HIV`, use the HIV HXB2 complete genome for alignment.

Customized target configuration

To define your own target configuration, create a JSON file. The root child `genes` contains a list of coding regions, with `begin` and `end`, the name of the gene, and a list of drug resistant mutations. Each DRM consists of its name and the positions it targets. The `drms` field is optional. If provided, the `referenceSequence` is used to call mutations, otherwise it will be tested against the major codon. All indices are with respect to the provided alignment space, 1-based, begin-inclusive and end-exclusive `[]`.

Target configuration: Example 1- A customized `json` target configuration file named `my_customized_hiv.json`:

```
{
  "genes": [
    {
      "begin": 2550,
      "drms": [
        {
          "name": "fancy drug",
          "positions": [ "M41L" ]
        }
      ],
      "end": 2700,
      "name": "Reverse Transcriptase"
    }
  ],
  "referenceName": "my seq",
  "referenceSequence": "TGGAAGGGCT..."
}
```

```

"version": "Free text to version your config files"
"databaseVersion": "DrugDB version x.y.z (last updated YYYY-MM-DD)"
}

```

Run with a customized target configuration using the `--config` option:

```

juliet --config my_customized_hiv.json data.align.bam patientZero.html

```

Valid formats for DRMs/positions

103	Only the reference position.
M130	Reference amino acid and reference position.
M103L	Reference aa, reference position, mutated aa.
M103LKA	Reference aa, reference position, list of possible mutated aas.
103L	Reference position and mutated aa.
103LG	Reference position and list mutated aas.

Missing amino acids are processed as wildcard (*).

Example:

```

{ "name": "ATV/r", "positions": [ "V32I", "L33", "46IL",
  "I54VTALM", "V82ATFS", "84" ] }

```

Target configuration: Example 2 - BCR-ABL:

For BCR-ABL, using the ABL1 gene with the following [reference](#) NM_005157.5, a typical target configuration looks like this:

```

{
  "genes": [
    {
      "name": "ABL1",
      "begin": 193,
      "end": 3585,
      "drms": [
        {
          "name": "imatinib",
          "positions": [
            "T315AI", "Y253H", "E255KV", "V299L", "F317AICLV", "F359CIV" ]
        },
        {
          "name": "dasatinib",
          "positions": [ "T315AI", "V299L", "F317AICLV" ]
        },
        {
          "name": "nilotinib",
          "positions": [ "T315AI", "Y253H", "E255KV", "F359CIV" ]
        },
        {
          "name": "bosutinib",
          "positions": [ "T315AI" ]
        }
      ]
    }
  ],
  "referenceName": "NM_005157.5",
  "referenceSequence": "TTAACAGGCGCGTCCC..."
}

```

No target configuration

If **no** target configuration is specified, either make sure that the sequence is in-frame, or specify the region of interest to mark the correct reading frame, so that amino acids are correctly translated. The output is labeled with `unknown` as the gene name:

```
juliet data.align.bam patientZero.html
```

Phasing

The default mode is to call amino-acid/codon variants independently. Using the `--mode-phasing` option, variant calls from distinct haplotypes are clustered and visualized in the HTML output.

Protease										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
C G A	R	8	X	T G A	0.98	2931	MGI											

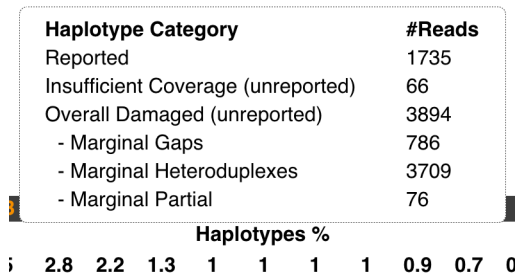
Reverse Transcriptase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A T G	M	41	L	T T G	0.99	2903	ABC + DDI + TDF + D4T + ZDV											
A A A	K	65	R	A G A	1	2577	3TC + FTC + ABC + DDI + TDF + D4T											
G G G	G	99	G	G G T	0.72	2907												
T T A	L	100	F	T T T	0.85	2819	MGI											
T A T	Y	181	C	T G T	0.95	2939	NVP + EFV + ETR + RPV											
G G A	G	190	A	G C A	1	2941	MGI + NVP + EFV + ETR + RPV											
A C C	T	215	Y	T A C	0.88	2940	ABC + DDI + TDF + D4T + ZDV											

Integrase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A A A	K	188	K	A A G	0.92	2923	MGI											

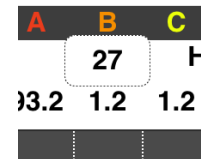
- The row-wise variant calls are "transposed" onto per-column haplotypes. Each haplotype has an ID: `[A-Z]{1}[a-z]?`.
- For each variant, colored boxes in this row mark haplotypes that contain this variant.
- Colored boxes per haplotype/column indicate variants that co-occur. Wild type (no variant) is represented by plain dark gray. A color palette helps to distinguish between columns.
- The JSON variant positions has an additional `haplotype_hit` boolean array with the length equal to the number of haplotypes. Each entry indicates if that variant is present in the haplotype. A haplotype block under the root of the JSON file contains counts and read names. The order of those haplotypes matches the order of all `haplotype_hit` arrays.

There are two types of tooltips in the haplotype section of the table.

The first tooltip is for the **Haplotypes %** and shows the number of reads that count towards (A) Actually reported haplotypes, (B) Haplotypes that have less than 10 reads and are not being reported, and (C) Haplotypes that are not suitable for phasing. Those first three categories are mutually exclusive and their sum is the total number of reads going into *juliet*. For (C), the three different marginals provide insights into the sample quality; as they are marginals, they are not exclusive and can overlap. The following image shows a sample with bad PCR conditions:



The second type of tooltip is for each haplotype percentage and shows the number of reads contributing to this haplotype:



**Microbial
Genome
Analysis**

The Microbial Genome Analysis application is powered by the IPA HiFi genome assembler and includes a base modification detection feature performed after the assembly.

Workflow of the Microbial Genome Analysis application

The workflow consists of several steps around 2 main stages:

1. **Chromosomal stage:** Assemble large contigs using IPA, the HiFi genome assembly tool.
2. Separate reads that were used for accurate large contigs from all other reads.
3. **Plasmid stage:** Assemble plasmids (using IPA) from the reads separated in the previous step.
4. De-duplicate plasmids.
5. Collect all contigs into a single FASTA file.
6. Rotate circular contigs.
7. Align the input Data Set to the assembled contigs.
8. Polish assembled contigs using `Racon`.
9. Perform base modification detection.

The application accepts HiFi XML Data Sets as input, and has an embedded downsampling feature:

- If the genome size and the desired coverage are specified, **both** stages of assembly are downsampled, as with the Genome Assembly application.
- Otherwise, the full coverage is used.

The embedded downsampling feature is **not** applied to the alignment stage; **all** input reads will be aligned against the assembled contigs.

Usage

The Microbial Genome Analysis application is run using the `pbcrumwell` run command, with the `pb_microbial_analysis` parameter to specify the application. See “[pbcrumwell](#)” on page 75 for details.

To view information on the available Microbial Genome Analysis options, enter:

```
pbcrumwell show-workflow-details pb_microbial_analysis
```

The minimum command needed to run the workflow requires the input Data Set.

The following example performs assembly and base modification detection using an input XML Data Set, and uses all default settings, including 1 CPU:

```
pbcrumwell run pb_microbial_analysis -e <input.consensusreadset.xml>
```


Note: To specify different task options on the command line, consider the following example:

```
pbcbromwell run pb_microbial_analysis \
-e <input.consensusreadset.xml> \
--task-option ipa2_genome_size=0 \
--task-option ipa2_downsampled_coverage=0 \
--task-option microasm_plasmid_contig_len_max=300000
--task-option ipa2_cleanup_intermediate_files=True \
--task-option dataset_filters="" \
--task-option filter_min_qv=20 \
--nproc 8
```

The default options for this application should work well for any genome type.

As microbes are relatively small, we rarely find much advantage to using more than 4 threads, or more than 2 concurrent jobs.

Microbial Genome Analysis parameters

Option	Default value	Description
-e, --eid_ccs	NONE	This is a SMRT Link-specific input parameter and supports only PacBio Consensusreadset XML files as input.
--task-option reads	NONE	Optional parameter, required if -e <input> is not specified. Supports multiple input formats: FASTA, FASTQ, BAM, XML, FOFN and gzipped versions of FASTA/FASTQ.
--task-option ipa2_genome_size	10M	The approximate number of base-pairs expected in the chromosomal genome. Used only for downsampling in the assembly stages (that is, not in polishing). If value <=0, then downsampling is off. Default: 10M (10 Mega basepairs). Note: ipa2_genome_size is currently the only option that accepts a metric suffix. Commas and decimals are not accepted anywhere. ipa2_genome_size actually accepts a String, which can be an integer followed by one of these metric suffixes: k/M/G. For example: 4500k means "4,500 kilobases" or "4,500,000". M stands for Mega and G stands for Giga.
--task-option ipa2_downsampled_coverage	100	The maximum coverage after downsampling with respect to the estimated genome_size. (The default genome_size was chosen to be larger than most realistic microbes.) If value <=0, then downsampling is off. Default: 100 Example: If your genome is 1G in length, and you specify ipa2_genome_size=2G, you have over-estimated by 2x. If you also specify ipa2_downsampled_coverage=100, your data will be downsampled to 200x coverage, simply because of the over-estimate. The cost of extra coverage is greater runtime. The defaults are usually fine.

Option	Default value	Description
--task-option ipa2_advanced_options_chrom	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) config_block_size = 100; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75
--task-option ipa2_advanced_options_plasmid	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) con-fig_block_size = 100; con-fig_ovl_filter_opt = --max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 500 --gapFilt --minDepth 4 --idt-stage2 98; con-fig_ovl_min_len = 500; con-fig_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --min-map-len 500 --min-anchor-span 500 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75 --smart-hit-per-target --secondary-min-ovl-frac 0.05; con-fig_layout_opt = -allow-circular;
--task-option ipa2_cleanup_intermediate_files	TRUE	Removes intermediate files from the run directory to save space.
--task-option microasm_plasmid_contig_len_max	300000	After the chromosomal stage, in task filter_draft_contigs, separates long contigs (presumed to be chromosomal) from shorter contigs (to be re-assembled in the plasmid stage). Then, after the plasmid stage, in dedup_plasmids, contigs less than this value are ignored. The default value is usually fine.
--task-option microasm_run_secondary_polish	TRUE	Specifies that an additional round of polishing will be applied after the two stages of assembly have completed.
--task-option run_basemods	TRUE	Specifies that base modification analysis be performed.
--task-option kineticstools_identify_mods	m4C,m6A	Specify the base modifications to identify, in a comma-separated list.
--task-option kineticstools_p_value	0.001	Specifies the probability value cutoff for detecting base modifications.
--task-option motif_min_score	35	Specifies the minimum QMod score used to identify a motif.
--task-option motif_min_fraction	0.30	Specifies the minimum methylated fraction to identify a motif.
--task-optionrun_find_motifs	TRUE	Specifies that motif-finding be performed.
--task-option dataset_filters	NONE	(General pbcromwell option) A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
--task-option filter_min_qv	20	(General pbcromwell option) Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.

Option	Default value	Description
--task-option ipa2_advanced_options_chrom	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) config_block_size = 100; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75
--task-option ipa2_advanced_options_plasmid	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) con-fig_block_size = 100; con-fig_ovl_filter_opt = --max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 500 --gapFilt --minDepth 4 --idt-stage2 98; con-fig_ovl_min_len = 500; con-fig_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --min-map-len 500 --min-anchor-span 500 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75 --smart-hit-per-target --secondary-min-ovl-frac 0.05; con-fig_layout_opt = -allow-circular;
--task-option ipa2_cleanup_intermediate_files	TRUE	Removes intermediate files from the run directory to save space.
--task-option microasm_plasmid_contig_len_max	300000	After the chromosomal stage, in task filter_draft_contigs, separates long contigs (presumed to be chromosomal) from shorter contigs (to be re-assembled in the plasmid stage). Then, after the plasmid stage, in dedup_plasmids, contigs less than this value are ignored. The default value is usually fine.
--task-option microasm_run_secondary_polish	TRUE	Specifies that an additional round of polishing will be applied after the two stages of assembly have completed.
--task-option run_basemods	TRUE	Specifies that base modification analysis be performed.
--task-option kineticstools_identify_mods	m4C,m6A	Specify the base modifications to identify, in a comma-separated list.
--task-option kineticstools_p_value	0.001	Specifies the probability value cutoff for detecting base modifications.
--task-option motif_min_score	35	Specifies the minimum QMod score used to identify a motif.
--task-option motif_min_fraction	0.30	Specifies the minimum methylated fraction to identify a motif.
--task-optionrun_find_motifs	TRUE	Specifies that motif-finding be performed.
--task-option dataset_filters	NONE	(General pbcromwell option) A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
--task-option filter_min_qv	20	(General pbcromwell option) Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.

Option	Default value	Description
<code>--task-option log_level</code>	INFO	Specifies the logging (verbosity) level for tools which support this feature. Values are: [TRACE, DEBUG, INFO, WARN, FATAL]
<code>--task-option nproc</code>	1	Specified the number of CPU threads to use.
<code>--task-option max_nchunks</code>	40	Specifies the maximum number of chunks per task.
<code>--task-option tmp_dir</code>	/tmp	Specifies an optional temporary directory for used by several subtools, such as <code>sort</code> .

Input files

- *.bam file containing PacBio data.
- *.xml file containing PacBio data.

Output files

- `final_assembly.fasta`: File containing all assembled contigs, rotated.
- `assembly.rotated.polished.renamed.fsa`: File for NCBI, but otherwise identical to `final_assembly.fasta`; only the headers are changed.
- `pb_microbial_analysis.motifs_csv`: File containing motifs and modifications.

Advanced assembly options

Note: Advanced parameters should be rarely modified. Advanced parameters for the Microbial Genome Assembly workflow are the **same** as those for the Genome Assembly workflow. See [“Advanced parameters” on page 44](#) for details.

Advanced parameters specified on the command line:

- Are in the form of `key = value` pairs.
- Each pair is separated by a semicolon (;) character.
- The full set of advanced parameters is surrounded by **one** set of double quotes.
- The specified value of a parameter **overwrites** the default options for that key. **All** desired options of that parameter must be explicitly listed, not just the ones which should change from the default.
- Setting an empty value **clears** the parameter; it does **not** reset the value back to default.

Example:

```
--task-option ipa2_advanced_options_chrom="config_seeddb_opt=-k
28;config_block_size=2048"
```

motifMaker The `motifMaker` tool identifies motifs associated with DNA modifications in prokaryotic genomes. Modified DNA in prokaryotes commonly arises from restriction-modification systems that methylate a specific base in a specific sequence motif. The canonical example is the m6A methylation of adenine in GATC contexts in *E. coli*. Prokaryotes may have a very large number of active restriction-modification systems present, leading to a complicated mixture of sequence motifs.

PacBio SMRT sequencing is sensitive to the presence of methylated DNA at single base resolution, via shifts in the polymerase kinetics observed in the real-time sequencing traces. For more background on modification detection, see [here](#).

Algorithm

Existing motif-finding algorithms such as MEME-chip and YMF are sub-optimal for this case for the following reasons:

- They search for a **single** motif, rather than attempting to identify a complicated mixture of motifs.
- They generally don't accept the notion of aligned motifs - the input to the tools is a window into the reference sequence which can contain the motif at any offset, rather than a single center position that is available with kinetic modification detection.
- Implementations generally either use a Markov model of the reference (MEME-chip), or do exact counting on the reference, but place restrictions on the size and complexity of the motifs that can be discovered.

Following is a rough overview of the algorithm used by `motifMaker`: Define a motif as a set of tuples: (position relative to methylation, required base). Positions not listed in the motif are implicitly degenerate. Given a list of modification detections and a genome sequence, define the following objective function on motifs:

$$\text{Motif score}(\text{motif}) = (\# \text{ of detections matching motif}) / (\# \text{ of genome sites matching motif}) * (\text{Sum of log-pvalue of detections matching motif}) = (\text{fraction methylated}) * (\text{sum of log-pvalues of matches})$$

Then, search (close to exhaustively) through the space of all possible motifs, progressively testing longer motifs using a branch-and-bound search. The “fraction methylated” term must be less than 1, so the maximum achievable score of a child node is the sum of scores of modification hits in the current node, allowing pruning of all search paths whose maximum achievable score is less than the best score discovered so far.

Usage

Run the `find` command, and pass the reference FASTA and the `modifications.gff.gz` file output by the PacBio modification detection workflow.

The `reprocess` subcommand annotates the GFF file with motif information for better genome browsing.

```
MotifMaker [options] [command] [command options]
```

`find` Command: Run motif-finding.

```
find [options]
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>* -f, --fasta</code>	Reference FASTA file.
<code>* -g, --gff</code>	Modifications.gff or .gff.gz file.
<code>-m, --minScore</code>	Specifies the minimum Qmod score to use in motif finding. (Default = 40.0)
<code>* -o, --output</code>	Outputs motifs.csv file.
<code>-x, --xml</code>	Outputs motifs XML file.

`reprocess` Command: Update a `modifications.gff` file with motif information based on new Modification QV thresholds.

```
reprocess [options]
```

Options	Description
<code>-c, --csv</code>	Raw modifications.csv file.
<code>* -f, --fasta</code>	Reference FASTA file.
<code>* -g, --gff</code>	Modifications.gff or .gff.gz file.
<code>-m, --minFraction</code>	Specifies that only motifs above this methylated fraction are used. (Default = 0.75)
<code>-m, --motifs</code>	Motifs.csv file.
<code>* -o, --output</code>	Reprocessed modifications.gff file.

Output files

Using the `find` command:

- **Output CSV file:** This file has the same format as the standard **Fields included in motif_summary.csv** described in the [Methylome Analysis White Paper](#).

Using the `reprocess` command:

- **Output GFF file:** The format of the output file is the same as the input file, and is described in the [Methylome Analysis White Paper](#) under **Fields included in the modifications.gff file**.

pbccromwell The `pbccromwell` tool is PacBio's wrapper for the `cromwell` scientific workflow engine used to power SMRT Link. `pbccromwell` includes advanced utilities for interacting directly with a Cromwell server.

`pbccromwell` is designed primarily for running workflows distributed and supported by PacBio, but it is written to handle any valid WDL source (version 1.0), and is very flexible in how it takes input. PacBio workflows are expected to be found in the directory defined by the `SMRT_PIPELINE_BUNDLE_DIR` environment variable, which is automatically defined by the SMRT Link distribution.

Note that `pbccromwell` does **not** interact with SMRT Link services; to run a Cromwell workflow as a SMRT Link job, use `pbservice`. (For details, see ["pbservice" on page 91.](#))

Note: Interaction with the Cromwell server is primarily intended for developers and power users.

Usage:

```
pbccromwell run [-h] [--output-dir OUTPUT_DIR] [--overwrite] [-i INPUTS]
                [-e ENTRY_POINTS] [-n NPROC] [-c MAX_NCHUNKS]
                [--target-size TARGET_SIZE] [--queue QUEUE] [-o OPTIONS]
                [-t TASK_OPTIONS] [-b BACKEND] [-r MAX_RETRIES]
                [--tmp-dir TMP_DIR] [--config CONFIG] [--dry-run]
                [run, show-workflows, show-workflow-details, configure, submit, get-
                job, abort, metadata, show-running, wait]
```

Options	Description
<code>--output-dir OUTPUT_DIR</code>	Specifies the output directory for <code>cromwell</code> output. (Default = <code>cromwell_out</code>)
<code>--overwrite</code>	Overwrites the output directory, if it exists. (Default = <code>False</code>)
<code>-i INPUTS, --inputs INPUTS</code>	Specifies <code>cromwell</code> inputs and settings as JSON files. (Default = <code>None</code>)
<code>-e ENTRY_POINTS, --entry ENTRY_POINTS</code>	Specifies the entry point Data Set; may be repeated for workflows that take more than one input Data Set. Note that all PacBio workflows require at least one such entry point.
<code>-n NPROC, --nproc NPROC</code>	Specifies the number of processors per task. (Default = <code>1</code>)
<code>-c MAX_NCHUNKS, --max-nchunks MAX_NCHUNKS</code>	Specifies the maximum number of chunks per task. (Default = <code>None</code>)
<code>--target-size TARGET_SIZE</code>	Specifies the target chunk size. (Default = <code>None</code>)
<code>--queue QUEUE</code>	Specifies the cluster queue to use. (Default = <code>None</code>)
<code>-o OPTIONS, --options OPTIONS</code>	Specifies additional <code>cromwell</code> engine options, as a JSON file. (Default = <code>None</code>)
<code>-t TASK_OPTIONS, --task-option TASK_OPTIONS</code>	Specifies workflow- or task-level option as <code>key=value</code> strings, specific to the application. May be specified multiple times for multiple options. (Default = <code>[]</code>)
<code>-b BACKEND, --backend BACKEND</code>	Specifies the backend to use for running tasks. (Default = <code>None</code>)
<code>-r MAX_RETRIES, --maxRetries MAX_RETRIES</code>	Specifies the maximum number of times to retry a failing task. (Default = <code>1</code>)

Options	Description
<code>--tmp-dir TMP_DIR</code>	Specifies an optional temporary directory for <code>cromwell</code> tasks, which must exist on all compute hosts. (Default = None)
<code>--config CONFIG</code>	Specifies a Java configuration file for running <code>cromwell</code> . (Default = None)
<code>--dry-run</code>	Specifies that <code>cromwell</code> is not executed, but instead writes out final inputs and then exits. (Default = True)
<code>workflow</code>	Specifies the workflow ID (such as <code>pb_ccs</code> or <code>cromwell.workflows.pb_ccs</code> for PacBio workflows only) or a path to a Workflow Description Language (WDL) source file.

Enter `pbcrumwell {command} -h` for a command's options.

Examples:

Show available PacBio-developed workflows:

```
pbcrumwell show-workflows
```

Show details for a PacBio workflow:

```
pbcrumwell show-workflow-details pb_ccs
```

Generate `cromwell.conf` with HPC settings:

```
pbcrumwell configure --default-backend SGE --output-file cromwell.conf
```

Launch a PacBio workflow:

```
pbcrumwell run pb_ccs -e /path/to/movie.subreadset.xml --nproc 8 --config /full/ path/ to/cromwell.conf
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. (Default = None, writes to stdout.)
<code>--log-level=INFO</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = INFO)
<code>--debug</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = None)

`pbcrumwell run` Command: Run a Cromwell workflow. Multiple input modes are supported, including a `pbsmrtpipe`-like set of arguments (for PacBio workflows **only**), and JSON files already in the native Cromwell format.

All PacBio workflows have similar requirements to the `pbsmrtpipe` pipelines in previous SMRT Link versions:

1. One or more PacBio dataset XML entry points, usually a SubreadSet or ConsensusReadSet (`--entry-point <FILE>.`)
2. Any number of workflow-specific task options (`--task-option <OPTION>.`)
3. Engine options independent of the workflow, such as number of processors per task (`--nproc`), or compute backend (`--backend`).

Output is directed to a new directory: `--output-dir`, which defaults to `cromwell_out`. This includes final inputs for the Cromwell CLI, and subdirectories for logs (workflow and task outputs), links to output files, and the Cromwell execution itself, which has a complex nested directory structure. Detailed information about the workflow execution can be found in the file `metadata.json`, in the native Cromwell format.

Note that output file links do **not** include the individual resource files of Data Sets and reports (BAM files, index files, plot PNGs, and so on.) Follow the symbolic links to their real path (for example using `readlink -f`) to find report plots.

For further information about Cromwell, see the official documentation [here](#).

Workflow examples:

Run the CCS analysis:

```
pbccromwell run pb_ccs -e <SUBREADS> --nproc 8 --config /full/path/to/cromwell.conf
```

Run the Iso-Seq workflow, including mapping to a reference, and execute on SGE:

```
pbccromwell run pb_isoseq3 -e <SUBREADS> -e <PRIMERS> -e <REFERENCE> --nproc 8 -- config /full/path/to/cromwell.conf
```

Run a user-defined workflow:

```
pbccromwell run my_workflow.wdl -i inputs.json -o options.json --config /full/path/to/cromwell.conf
```

Set up input files but exit before starting Cromwell:

```
pbccromwell run pb_ccs -e <SUBREADS> --nproc 8 --dry-run
```

Print details about the named PacBio workflow, including input files and task options. **Note:** The prefix `cromwell.workflows.` is optional.

```
pbccromwell show-workflow-details pb_ccs
pbccromwell show-workflow-details cromwell.workflows.pb_ccs
```

`pbccromwell show-workflow-details` Command: Display details about a specified PacBio workflow, including input files and task options.

Usage:

```
pbccromwell show-workflow-details [-h] [--inputs-json INPUTS_JSON] workflow_id
```

Options	Description
<code>workflow_id</code>	Specifies the workflow ID, such as <code>pb_ccs</code> or <code>cromwell.workflows.pb_ccs</code> for PacBio workflows only .
<code>--inputs-json INPUTS_JSON</code>	Writes a JSON template containing workflow inputs to the specified file. (Default = None)

`pbccromwell configure` Command: Generate the Java configuration file used by `cromwell` to define backends and other important engine options that **cannot** be set at runtime. You can pass this to `pbccromwell run` using the `--config` argument.

Usage:

```
pbccromwell configure [-h] [--port PORT]
                        [--local-job-limit LOCAL_JOB_LIMIT]
                        [--jms-job-limit JMS_JOB_LIMIT]
                        [--db-port DB_PORT] [--db-user DB_USER]
                        [--db-password DB_PASSWORD]
                        [--default-backend DEFAULT_BACKEND]
                        [--max-workflows MAX_WORKFLOWS]
                        [--output-file OUTPUT_FILE] [--timeout TIMEOUT]
                        [--cache] [--no-cache]
```

Options	Description
<code>--port PORT</code>	Specifies the port that <code>cromwell</code> should listen to. (Default = 8000)
<code>--local-job-limit LOCAL_JOB_LIMIT</code>	Specifies the maximum number of local jobs/tasks that can be run at once. (Default = 10)
<code>--jms-job-limit JMS_JOB_LIMIT</code>	Specifies the maximum number of jobs/tasks that can be submitted to the queueing system at one time. (Default = 500)
<code>--db-port DB_PORT</code>	Specifies the database port for <code>cromwell</code> to use; if undefined, database configuration is omitted. (Default = None)
<code>--db-user DB_USER</code>	Specifies the user name used to connect to Postgres. (Default = <code>smrtlink_user</code>)
<code>--db-password DB_PASSWORD</code>	Specifies the password used to connect to Postgres.
<code>--default-backend DEFAULT_BACKEND</code>	Specifies the default job execution backend. Choices are: Local, SGE, Slurm, PBS, or LSF. (Default = Local)
<code>--max-workflows MAX_WORKFLOWS</code>	Specifies the maximum number of workflows that <code>cromwell</code> can run at once. (Default = 100)
<code>--output-file OUTPUT_FILE</code>	Specifies the name of the output configuration file. (Default = <code>cromwell.conf</code>)
<code>--timeout TIMEOUT</code>	Specifies the time to wait for task completion before checking the cluster job status. (Default = 600)

Options	Description
--cache	Enables call caching. (Default = True)
--no-cache	Disables call caching. (Default = True)

pbindex The `pbindex` tool creates an index file that enables random access to PacBio-specific data in BAM files.

Usage

`pbindex <input>`

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.

Input file

- `*.bam` file containing PacBio data.

Output file

- `*.pbi` index file, with the same prefix as the input file name.

pbmarkdup The `pbmarkdup` tool marks PCR duplicates in CCS reads Data Sets from amplified libraries. PCR duplicates are different reads that arose from amplifying a single-source molecule. `pbmarkdup` can also optionally remove the duplicate reads.

Note: `pbmarkdup` only works with CCS reads, **not** with Subreads.

`pbmarkdup` uses a reference-free comparison method. Duplicates are identified as pairs of reads that:

1. Have the same length - within 10 bp, and
2. Have high percent identity alignments at the molecule ends at >98% identity of the first and last 250 bp.

Clusters are formed from sets of two or more duplicate reads, and a single read is selected as the representative of each cluster. Other reads in the cluster are considered duplicates.

How are duplicates marked?

In FASTA and FASTQ formats, reads from duplicate clusters have annotated names. The following is a FASTA example:

```
>m64013_191117_050515/67104/ccs DUPLICATE=m64013_191117_050515/3802014/ccs DS=2
```

This shows a marked duplicate read `m64013_191117_050515/67104/ccs` that is a duplicate of `m64013_191117_050515/3802014/ccs` in a cluster with 2 reads (`DS` value). Accordingly, the following is the read selected as the representative of the molecule:

```
>m64013_191117_050515/3802014/ccs DS=2
```

In BAM format, duplicate reads are flagged with `0x400`. The read-level tag `ds` provides the number of reads in a cluster (like the `DS` attribute described above for FASTA/FASTQ), and the `du` tag provides the name of the representative read (like the `DUPLICATE` attribute described above for FASTA/FASTQ).

Usage

```
pbmarkdup [options] <INFILE.bam|xml|fa|fq|fofn> <OUTFILE.bam|xml|fa.gz|fq.gz>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file</code>	Logs to a file, instead of <code>stderr</code> .

Options	Description																
<code>--log-level</code>	<p>Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL] (Default = WARN)</p> <p><code>--log-level INFO</code> produces a summary report such as:</p> <table><tr><th>LIBRARY</th><th>READS</th><th>UNIQUE MOLECULES</th><th>DUPLICATE READS</th></tr><tr><td><Unnamed></td><td>25000</td><td>24948 (99.8%)</td><td>52 (0.2%)</td></tr><tr><td>SS-lib</td><td>496</td><td>493 (99.4%)</td><td>3 (0.6%)</td></tr><tr><td>TOTAL</td><td>25496</td><td>25441 (99.8%)</td><td>55 (0.2%)</td></tr></table>	LIBRARY	READS	UNIQUE MOLECULES	DUPLICATE READS	<Unnamed>	25000	24948 (99.8%)	52 (0.2%)	SS-lib	496	493 (99.4%)	3 (0.6%)	TOTAL	25496	25441 (99.8%)	55 (0.2%)
LIBRARY	READS	UNIQUE MOLECULES	DUPLICATE READS														
<Unnamed>	25000	24948 (99.8%)	52 (0.2%)														
SS-lib	496	493 (99.4%)	3 (0.6%)														
TOTAL	25496	25441 (99.8%)	55 (0.2%)														
<code>-j, --num-threads</code>	<p>Specifies the number of threads to use, 0 means autodetection. (Default = 0)</p>																

Duplicate marking options	Description
<code>--cross-library, -x</code>	Identifies duplicate reads across sequencing libraries. Libraries are specified in the BAM read group LB tag.

Output options	Description
<code>-rmdup, -r</code>	Excludes duplicates from OUTFILE. (This is redundant when <code>--dup-file</code> is specified.)
<code>--dup-file FILE</code>	Stores duplicate reads in an extra file other than OUTFILE. The format of this file can be different from the output file.
<code>--clobber, -f</code>	Overwrites OUTFILE if it exists.

Input files

CCS reads from one or multiple movies in any of the following formats:

- PacBio BAM (.ccs.bam)
- PacBio dataset (.dataset.xml)
- File of File Names (.fofn)
- FASTA (.fasta, .fasta.gz)
- FASTQ (.fastq, .fastq.gz)

Output files

CCS reads with duplicates marked in a format inferred from the file extension:

- PacBio BAM (.ccs.bam)
- PacBio dataset (.dataset.xml), which also generates a corresponding BAM file.
- FASTA (.fasta.gz)
- FASTQ (.fastq.gz)

Allowed input/output combinations

Input File	Output BAM	Output Dataset	Output FASTQ	Output FASTA
Input BAM	Allowed	Allowed	Allowed	Allowed
Input Dataset	Allowed	Allowed	Allowed	Allowed
Input FASTQ	Not Allowed	Not Allowed	Allowed	Allowed
Input FASTA	Not Allowed	Not Allowed	Not Allowed	Allowed

Examples

Run on a single movie:

```
pbmarkdup movie.ccs.bam output.bam
```

Run on multiple movies:

```
pbmarkdup movie1.fasta movie2.fasta output.fasta
```

Run on multiple movies and output duplicates in separate file:

```
pbmarkdup movie1.ccs.bam movie2.fastq uniq.fastq --dup-file dups.fasta
```

pbmm2 The `pbmm2` tool aligns native PacBio data, outputs PacBio BAM files, and is a SMRT `minimap2` wrapper for PacBio data.

`pbmm2` supports native PacBio input and output, provides sets of recommended parameters, generates sorted output on-the-fly, and post-processes alignments. Sorted output can be used directly for polishing using `GenomicConsensus`, if BAM has been used as input to `pbmm2`.

`pbmm2` adds the following SAM tag to each aligned record:

- `mg`, stores gap-compressed alignment identity, defined as $nM / (nM + nMM + nInsEvents + nDelEvents)$.

Usage

```
pbmm2 <tool>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.

`index` Command: Indexes references and stores them as `.mmi` files. Indexing is optional, but recommended if you use the same reference with the same `--preset` multiple times.

Usage:

```
pbmm2 index [options] <ref.fa|xml> <out.mmi>
```

Input file

- `*.fasta`, `*.fa` file containing reference contigs or `*.referenceset.xml`.

Output file

- `out.mmi` (`minimap2` index file.)

Notes:

- You can use existing `minimap2 .mmi` files with `pbmm2 align`.
- If you use an index file, you **cannot** override parameters `-k`, `-w`, nor `-u` in `pbmm2 align`.
- The `minimap2` parameter `-H` (homopolymer-compressed k-mer) is always on for SUBREAD and UNROLLED presets, and can be disabled using `-u`.

Options	Description
<code>--preset</code>	Specifies the alignment mode: <ul style="list-style-type: none"> • "SUBREAD" -k 19 -w 10 • "CCS" -k 19 -w 10 -u • "ISOSEQ" -k 15 -w 5 -u • "UNROLLED" -k 15 -w 15 The option is not case-sensitive. (Default = SUBREAD)
<code>-k</code>	Specifies the k-mer size, which cannot be larger than 28. (Default = -1)
<code>-w</code>	Specifies the Minimizer window size. (Default = -1)
<code>-u, --no-kmer-compression</code>	Disables homopolymer-compressed k-mer. (Compression is on by default for the SUBREAD and UNROLLED presets.)
<code>--short-sa-cigar</code>	Populates the SA tag with the short cigar representation.

align Command: Aligns PacBio reads to reference sequences. The output argument is optional; if not provided, the BAM output is streamed to stdout.

Usage:

```
pbmm2 align [options] <ref.fa|xml|mml> <in.bam|xml|fa|fq> [out.aligned.bam|xml]
```

Input files

- *.fasta file containing reference contigs, or *.referenceset.xml, or *.mml index file.
- *.bam, *.subreadset.xml, *.consensusreadset.xml, *.transcriptset.xml, *.fasta, *.fa, *.fastq, or *.fastq file containing PacBio data.

Output files

- *.bam aligned reads in BAM format.
- *.alignmentset, *.consensusalignmentset.xml, or *.transcriptalignmentset.xml if XML output was chosen.

The following Data Set Input/output combinations are allowed:

SubreadSet > AlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.subreadset.xml hg38.movie.alignmentset.xml
```

ConsensusReadSet > ConsensusAlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.consensusreadset.xml
hg38.movie.consensusalignmentset.xml --preset CCS
```

TranscriptSet > TranscriptAlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.transcriptset.xml
hg38.movie.transcriptalignmentset.xml --preset ISOSEQ
```

FASTA/FASTQ input

In addition to native PacBio BAM input, reads can also be provided in FASTA and FASTQ formats.

Attention: The resulting output BAM file **cannot** be used as input into GenomicConsensus!

With FASTA/FASTQ input, the `--rg` option sets the read group. **Example:**

```
pbmm2 align hg38.fasta movie.Q20.fastq hg38.movie.bam --preset CCS --rg
 '@RG\tID:myid\tSM:mysample'
```

All three reference file formats `.fasta`, `.referenceset.xml`, and `.mmi` can be combined with FASTA/FASTQ input.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--chunk-size</code>	Processes <i>N</i> records per chunk. (Default = 100)
<code>--sort</code>	Generates a sorted BAM file.
<code>-m, --sort-memory</code>	Specifies the memory per thread for sorting. (Default = 768M)
<code>-j, --alignment-threads</code>	Specifies the number of threads used for alignment. 0 means autodetection. (Default = 0)
<code>-J, --sort-threads</code>	Specifies the number of threads used for sorting. 0 means 25% of <code>-j</code> , with a maximum of 8. (Default = 0)
<code>--sample</code>	Specifies the sample name for all read groups. Defaults, in order of precedence: A) SM field in the input read group B) Biosample name C) Well sample name D) "UnnamedSample".
<code>--rg</code>	Specifies the read group header line such as <code>'@RG\tID:xyz\tSM:abc'</code> . Only for FASTA/Q inputs.
<code>-y, --min-gap-comp-id-perc</code>	Specifies the minimum gap-compressed sequence identity, in percent. (Default = 70)
<code>-l, --min-length</code>	Specifies the minimum mapped read length, in base pairs. (Default = 50)
<code>-N, --best-n</code>	Specifies the output at maximum <i>N</i> alignments for each read. 0 means no maximum. (Default = 0)
<code>--strip</code>	Removes all kinetic and extra QV tags. The output cannot be polished.
<code>--split-by-sample</code>	Specifies one output BAM file per sample.
<code>--no-bai</code>	Omits BAI index file generation for sorted output.
<code>--unmapped</code>	Specifies that unmapped records be included in the output.
<code>--median-filter</code>	Picks one read per ZMW of median length.
<code>--zmw</code>	Processes ZMW Reads; <code>subreadset.xml</code> input is required. This activates the UNROLLED preset.

Options	Description
<code>--hqregion</code>	Processes the HQ region of each ZMW; <code>subreadset.xml</code> input is required. This activates the UNROLLED preset.

Parameter set options and overrides	Description
<code>--preset</code>	Specifies the alignment mode: <ul style="list-style-type: none"> • "SUBREAD" <code>-k 19 -w 10 -o 5 -O 56 -e 4 -E 1 -A 2 -B 5 -z 400 -Z 50 -r 2000 -L 0.5</code> • "CCS" <code>-k 19 -w 10 -u -o 5 -O 56 -e 4 -E 1 -A 2 -B 5 -z 400 -Z 50 -r 2000 -L 0.5</code> • "ISOSEQ" <code>-k 15 -w 5 -u -o 2 -O 32 -e 1 -E 0 -A 1 -B 2 -z 200 -Z 100 -C 5 -r 200000 -G 200000 -L 0.5</code> • "UNROLLED" <code>-k 15 -w 15 -o 2 -O 32 -e 1 -E 0 -A 1 -B 2 -z 200 -Z 100 -r 2000 -L 0.5</code> (Default = SUBREAD)
<code>-k</code>	Specifies the k-mer size, which cannot be larger than 28. (Default = -1)
<code>-w</code>	Specifies the Minimizer window size. (Default = -1)
<code>-u, --no-kmer-compression</code>	Disables homopolymer-compressed k-mer. (Compression is on by default for the SUBREAD and UNROLLED presets.)
<code>-A</code>	Specifies the matching score. (Default = -1)
<code>-B</code>	Specifies the mismatch penalty. (Default = -1)
<code>-z</code>	Specifies the Z-drop score. (Default = -1)
<code>-Z</code>	Specifies the Z-drop inversion score. (Default = -1)
<code>-r</code>	Specifies the bandwidth used in chaining and DP-based alignment. (Default = -1)
<code>-o, --gap-open-1</code>	Specifies the gap open penalty 1. (Default = -1)
<code>-O, --gap-open-2</code>	Specifies the gap open penalty 2. (Default = -1)
<code>-e, --gap-extend-1</code>	Specifies the gap extension penalty 1. (Default = -1)
<code>-E, --gap-extend-2</code>	Specifies the gap extension penalty 2. (Default = -1)
<code>-L, --lj-min-ratio</code>	Specifies the long join flank ratio. (Default = -1)
<code>-G</code>	Specifies the maximum intron length; this changes <code>-r</code> . (Default = -1)
<code>-C</code>	Specifies the cost for a non-canonical GT-AG splicing. (Default = -1)
<code>--no-splice-flank</code>	Specifies that you do not prefer splicing flanks GT-AG.

Examples:

Generate an index file for reference and reuse it to align reads:

```
pbmm2 index ref.fasta ref.mmi
pbmm2 align ref.mmi movie.subreads.bam ref.movie.bam
```

Align reads and sort on-the-fly, with 4 alignment and 2 sort threads:

```
pbmm2 align ref.fasta movie.subreads.bam ref.movie.bam --sort -j 4 -J 2
```

Align reads, sort on-the-fly, and create a PBI:

```
pbmm2 align ref.fasta movie.subreadset.xml ref.movie.alignmentset.xml --sort
```

Omit the output file and stream the BAM output to `stdout`:

```
pbmm2 align hg38.mmi movie1.subreadset.xml | samtools sort > hg38.movie1.sorted.bam
```

Align the CCS reads fastq input and sort the output:

```
pbmm2 align ref.fasta movie.Q20.fastq ref.movie.bam --preset CCS --sort --rg
'@RG\tID:myid\tSM:mysample'
```

Alignment parallelization

The number of alignment threads can be specified using the `-j`, `--alignment-threads` option. If **not** specified, the maximum number of threads are used, minus one thread for BAM I/O and minus the number of threads specified for sorting.

Sorting

Sorted output can be generated using the `--sort` option.

- By default, 25% of threads specified with the `-j` option (Maximum = 8) are used for sorting.
- To override the default percentage, the `-J`, `--sort-threads` option defines the explicit number of threads used for on-the-fly sorting. The memory allocated per sort thread is defined using the `-m`, `--sort-memory` option, accepting suffixes M,G.

Benchmarks on human data show that 4 sort threads are recommended, but that no more than 8 threads can be effectively leveraged, even with 70 cores used for alignment. We recommend that you provide more memory to **each** of a **few** sort threads to avoid disk I/O pressure, rather than providing less memory to each of many sort threads.

What are parameter sets and how can I override them?

Per default, `pbmm2` uses recommended parameter sets to simplify the multitudes of possible combinations. See the available parameter sets in the option table shown earlier.

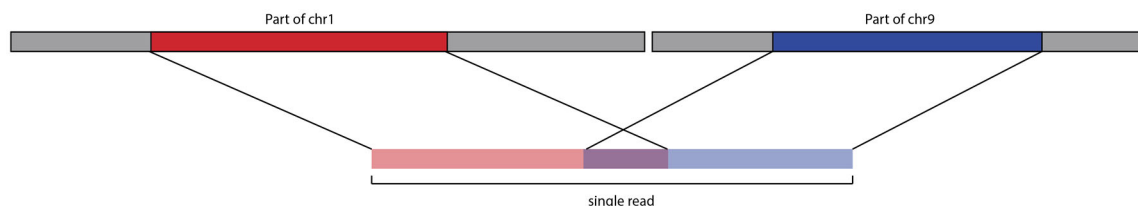
What other special parameters are used implicitly?

We implicitly use the following `minimap2` parameters:

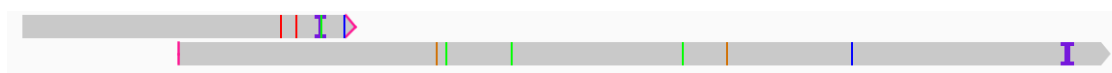
- Soft clipping with `-Y`.
- Long cigars for tag `CG` with `-L`.
- `X/=` cigars instead of `M` with `--eqx`.
- No overlapping query intervals with repeated matches trimming.
- No secondary alignments are produced using the `--secondary=no` option.

What is repeated matches trimming?

A repeated match occurs when the same query interval is shared between a primary and supplementary alignment. This can happen for translocations, where breakends share the same flanking sequence:



And sometimes, when a LINE gets inserted, the flanks are duplicated, leading to complicated alignments, where we see a split read sharing a duplication. The inserted region itself, mapping to a random other LINE in the reference genome, may also share sequence similarity to the flanks:



To get the best alignments, `minimap2` decides that two alignments may use up to 50% (default) of the same query bases. This does **not** work for PacBio, as `pbbmm2` requires that a single base may never be aligned twice. `Minimap2` offers a feature to enforce a query interval overlap to 0%. If a query interval gets used in two alignments, one or both get flagged as secondary and get filtered. This leads to yield loss, and more importantly, missing SVs in the alignment.

Papers (such as [this](#)) present dynamic programming approaches to find the optimal split to uniquely map query intervals, while maximizing alignment scores. We don't have per base alignment scores available, thus our approach is much simpler. We align the read, find overlapping query intervals, determine one alignment to be maximal reference-spanning, then trim all others. By trimming, `pbbmm2` rewrites the cigar and the reference coordinates on-the-fly. This allows us to increase the number of mapped bases, which slightly reduces mapped concordance, but boosts SV recall rate.

How can I set the sample name?

You can override the sample name (`SM` field in the `RG` tag) for **all** read groups using the `--sample` option. If not provided, sample names derive from the Data Set input using the following order of precedence: A) `SM` field in the input read group B) Biosample name C) Well sample name D) `UnnamedSample`. If the input is a BAM file and the `--sample` option was **not** used, the `SM` field is populated with `UnnamedSample`.

Can I split output by sample name?

Yes, the `--split-by-sample` option generates one output BAM file per sample name, with the sample name as the file name prefix, if there is more than one aligned sample name.

Can I remove all those extra per-base and per-pulse tags?

Yes, the `--strip` option removes the following extraneous tags if the input is BAM: `dq`, `dt`, `ip`, `iq`, `mq`, `pa`, `pc`, `pd`, `pe`, `pg`, `pm`, `pq`, `pt`, `pv`, `pw`, `px`, `sf`, `sq`, `st`. Note that the resulting output BAM file **cannot** be used as input into `GenomicConsensus`.

Where are the unmapped reads?

Per default, unmapped reads are omitted. You can add them to the output BAM file using the `--unmapped` option.

Can I output at maximum the N best alignments per read?

Use the option `-N`, `--best-n`. If set to 0, (the default), maximum filtering is disabled.

Is there a way to only align one subread per ZMW?

Using the `--median-filter` option, only the subread closest to the median subread length per ZMW is aligned. Preferably, full-length subreads flanked by adapters are chosen.

pbserve The `pbserve` tool performs a variety of useful tasks within SMRT Link.

- To get help for `pbserve`, use `pbserve -h`.
- To get help for a specific `pbserve` command, use `pbserve <command> -h`.

Note: `pbserve` requires authentication when run from a remote host, using the same credentials used to log in to the SMRT Link GUI. (This also routes all requests through HTTPS port 8243, so the services port is **not** required if authentication is used.) Access to services running on `localhost` will continue to work without authentication.

All `pbserve` commands include the following optional parameters:

Options	Description
<code>--host=http://localhost</code>	Specifies the server host. Override the default with the environmental variable <code>PB_SERVICE_HOST</code> .
<code>--port=8070</code>	Specifies the server port. Override the default with the environmental variable <code>PB_SERVICE_PORT</code> .
<code>--log-file LOG_FILE</code>	Writes the log to file. (Default = None, writes to stdout.)
<code>--log-level=INFO</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = INFO)
<code>--debug=False</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--quiet=False</code>	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
<code>--user USERNAME</code>	Specifies the user to authenticate as; this is required if the target host is anything other than <code>localhost</code> .
<code>--ask-pass</code>	Prompts the user to enter a password.
<code>--password PASSWORD</code>	Supplies the password directly. This exposes the password in the shell history (or log files), so this option is not recommended unless you are using a limited account without Unix login access.

`status` Command: Use to get system status.

```
pbserve status [-h] [--host HOST] [--port PORT]
               [--log-file LOG_FILE]
               [--log-level INFO]
               [--debug] [--quiet]
```

`import-dataset` Command: Import Local Data Set XML. The file location **must** be accessible from the host where the services are running; often on a shared file system.

```
pbserve import-dataset [-h] [--host HOST] [--port PORT]
                       [--log-file LOG_FILE]
                       [--log-level INFO]
                       [--debug] [--quiet]
                       xml_or_dir
```

Required	Description
xml_or_dir	Specifies a directory or XML file for the Data Set.

import-run Command: Create a SMRT Link Run Design and optional Auto Analysis jobs from a CSV file. This is equivalent to the Run Design CSV import feature in the SMRT Link UI; the resulting runs can then be edited in the UI or executed on-instrument.

```
pbservice import-run [-h][--host HOST] [--port PORT]
                    [--log-file LOG_FILE]
                    [--log-level INFO]
                    [--debug] [--quiet]
                    [--dry-run]
                    csv_file
```

Required	Description
csv_file	Specifies a Run Design CSV-format file.

Option	Description
--dry-run	Prints the generated run XML without POSTing to the server.

import-fasta Command: Import a FASTA file and convert to a ReferenceSet file. The file location **must** be accessible from the host where the services are running; often on a shared file system.

```
pbservice import-fasta [-h] --name NAME --organism ORGANISM --ploidy
PLOIDY [--block] [--host HOST] [--port PORT]
      [--log-file LOG_FILE]
      [--log-level INFO]
      [--debug] [--quiet]
      fasta_path
```

Required	Description
fasta_path	Path to the FASTA file to import.

Options	Description
--name	Specifies the name of the ReferenceSet to convert the FASTA file to.
--organism	Specifies the name of the organism.
--ploidy	Ploidy.
--block=False	Blocks during importing process.

run-analysis Command: Run a secondary analysis pipeline using an analysis.json file.

```
pbservice run-analysis [-h] [--host HOST] [--port PORT]
                      [--log-file LOG_FILE]
                      [--log-level INFO]
```



```

[--debug] [--quiet] [--block]
json_path

```

Required	Description
json_path	Path to the analysis.json file.

Options	Description
--block=False	Blocks during importing process.

emit-analysis-template Command: Output an analysis.json template to stdout that can be run using the run-analysis command.

```

pbservice emit-analysis-template [-h] [--log-file LOG_FILE]
                                [--log-level INFO]
                                [--debug] [--quiet]

```

get-job Command: Get a job summary by Job Id.

```

pbservice get-job [-h] [--host HOST] [--port PORT]
                  [--log-file LOG_FILE]
                  [--log-level INFO]
                  [--debug] [--quiet]
job_id

```

Required	Description
job_id	Job id or UUID.

get-jobs Command: Get job summaries by Job Id.

```

pbservice get-jobs [-h] [-m MAX_ITEMS] [--host HOST] [--port PORT]
                  [--log-file LOG_FILE]
                  [--log-level INFO]
                  [--debug] [--quiet]

```

Options	Description
-m=25, --max-items=25	Specifies the maximum number of jobs to get.

get-dataset Command: Get a Data Set summary by Data Set Id or UUID.

```

pbservice get-dataset [-h] [--host HOST] [--port PORT]
                     [--log-file LOG_FILE]
                     [--log-level INFO]
                     [--debug] [--quiet]
id_or_uuid

```

Required	Description
id_or_uuid	Data Set Id or UUID.

get-datasets Command: Get a Data Set list summary by Data Set type.

```

pbservice get-datasets [-h] [--host HOST] [--port PORT]
                      [--log-file LOG_FILE]
                      [--log-level INFO]
                      [--debug] [--quiet] [-m MAX_ITEMS]

```

```
[-t DATASET_TYPE]
```

Required	Description
<code>-t=subreads, --dataset-type=subreads</code>	Specifies the type of Data Set to retrieve: subreads, alignments, references, barcodes.

`delete-dataset` Command: Delete a specified Data Set.

Note: This is a "soft" delete - the database record is tagged as inactive so it won't display in any lists, but the files will **not** be removed.

```
pbservice delete-dataset [-h] [--host HOST] [--port PORT]
                        [--log-file LOG_FILE]
                        [--log-level INFO]
                        [--debug] [--quiet]
                        [ID]
```

Required	Description
ID	A valid Data Set ID, either UUID or integer ID, for the Data Set to delete.

Examples

To obtain system status, the Data Set summary, and the job summary:

```
pbservice status --host smrtlink-release --port 9091
```

To import a Data Set XML:

```
pbservice import-dataset --host smrtlink-release --port 9091 \
path/to/subreadset.xml
```

To obtain a job summary using the Job Id:

```
pbservice get-job --host smrtlink-release --port 9091 \
--log-level CRITICAL 1
```

To obtain Data Sets by using the Data Set type subreads:

```
pbservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t subreads
```

To obtain Data Sets by using the Data Set type alignments:

```
pbservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t alignments
```

To obtain Data Sets by using the Data Set type references:

```
pbservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t references
```

To obtain Data Sets by using the Data Set type barcodes:

```
pbservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t barcodes
```

To obtain Data Sets by using the Data Set UUID:

```
pbservice get-dataset --host smrtlink-alpha --port 8081 \  
--quiet 43156b3a-3974-4ddb-2548-bb0ec95270ee
```

pbsv `pbsv` is a structural variant caller for PacBio reads. It identifies structural variants and large indels (Default: ≥ 20 bp) in a sample or set of samples relative to a reference. `pbsv` identifies the following types of variants: Insertions, deletions, duplications, copy number variants, inversions, and translocations.

`pbsv` takes as input read alignments (BAM) and a reference genome (FASTA); it outputs structural variant calls (VCF).

Usage:

```
pbsv [-h] [--version] [--quiet] [--verbose]
      {discover,call}...
```

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of stdout.
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)
discover	Finds structural variant signatures in read alignments (BAM to SVSIG).
call	Calls structural variants from SV signatures and assign genotypes (SVSIG to VCF).

`pbsv discover` Command: Finds structural variant (SV) signatures in read alignments. The input read alignments must be sorted by chromosome position. Alignments are typically generated with `pbbmm2`. The output SVSIG file contains SV signatures.

Usage:

```
pbsv discover [options] <ref.in.bam|xml> <ref.out.svsig.gz>
```

Required	Description
ref.in.bam xml	Coordinate-sorted aligned reads in which to identify SV signatures.
ref.out.svsig.gz	Structural variant signatures output.

Options	Description
-h, --help	Displays help information and exits.
-s, --sample	Overrides sample name tag from BAM read group.
-q, --min-mapq	Ignores alignments with mapping quality < N. (Default = 20)
-m, --min-ref-span	Ignores alignments with reference length < N bp. (Default = 100)
-w, --downsample-window-length	Specifies a window in which to limit coverage, in base pairs. (Default = 10K)
-a, --downsample-max-alignments	Considers up to N alignments in a window; 0 means disabled. (Default = 100)

Options	Description
-r, --region	Limits discovery to this reference region: CHR CHR:START-END.
-l, --min-sv-sig-length	Ignores SV signatures with length < N bp. (Default = 7)
-b, --tandem-repeats	Specifies tandem repeat intervals for indel clustering, as an input BED file.
-k, --max-skip-split	Ignores alignment pairs separated by > N bp of a read or reference. (Default = 100)
-y, --min-gap-comp-id-perc	Ignores alignments with gap-compressed sequence identity < N%. (Default = 70)

pbsv call Command: Calls structural variants from SV signatures and assigns genotypes. The input SVSIG file is generated using **pbsv discover**. The output is structural variants in VCF format.

Usage:

```
pbsv call [options] <ref.fa|xml> <ref.in.svsig.gz|fofn>
<ref.out.vcf>
```

Required	Description
ref.fa xml	Reference FASTA file or ReferenceSet XML file against which to call variants.
ref.in.svsig.gz fofn	SV signatures from one or more samples. This can be either an SV signature SVSIG file generated by pbsv discover , or a FOFN of SVSIG files.
ref.out.vcf	Variant call format (VCF) output file.

Options	Description
-h, --help	Displays help information and exits.
-j, --num-threads	Specifies the number of threads to use, 0 means autodetection. (Default = 0)
-r, --region	Limits discovery to this reference region: CHR CHR:START-END
-t, --types	Calls these SV types: "DEL", "INS", "INV", "DUP", "BND", "CNV". (Default = "DEL, INS, INV, DUP, BND")
-m, --min-sv-length	Ignores variants with length < N bp. (Default = 20)
--max-ins-length	Ignores insertions with length > N bp. (Default = 15K)
--max-dup-length	Ignores duplications with length > N bp. (Default = 1M)
--cluster-max-length-perc-diff	Does not cluster signatures with difference in length > P%. (Default = 25)
--cluster-max-ref-pos-diff	Does not cluster signatures > N bp apart in the reference. (Default = 200)
--cluster-min-basepair-perc-id	Does not cluster signatures with base pair identity < P%. (Default = 10)
-x, --max-consensus-coverage	Limits to N reads for variant consensus. (Default = 20)
-s, --poa-scores	Scores POA alignment with triplet match, mismatch, gap. (Default = "1, -2, -2")
--min-realign-length	Considers segments with > N length for realignment. (Default = 100)

Options	Description
-A, --call-min-reads-all-samples	Ignores calls supported by < N reads total across samples. (Default = 3)
-O, --call-min-reads-one-sample	Ignores calls supported by < N reads in every sample. (Default = 3)
-S, --call-min-reads-per-strand-all-samples	Ignores calls supported by < N reads per strand total across samples. (Default = 1)
-B, --call-min-bnd-reads-all-samples	Ignores BND calls supported by < N reads total across samples. (Default = 2)
-P, --call-min-read-perc-one-sample	Ignores calls supported by < P% of reads in every sample. (Default = 20)
--preserve-non-acgt	Preserves non-ACGT in REF allele instead of replacing with N. (Default = false)
--hifi, --ccs	Uses options optimized for HiFi reads: -S 0 -P 10. (Default = False)
--gt-min-reads	Specifies the minimum supporting reads to assign a sample a non-reference genotype. (Default = 1)
--annotations	Annotates variants by comparing with sequences in FASTA. (Default annotations are ALU, L1, and SVA.)
--annotation-min-perc-sim	Annotates variant if sequence similarity > P%. (Default = 60)
--min-N-in-gap	Considers \geq N consecutive "N" bp as a reference gap. (Default = 50)
--filter-near-reference-gap	Flags variants < N bp from a gap as "NearReferenceGap". (Default = 1000)
--filter-near-contig-end	Flags variants < N bp from a contig end as "NearContigEnd". (Default = 1K)

Following is a typical SV analysis workflow starting from subreads:

1. Align PacBio reads to a reference genome, per movie:

Subreads BAM input:

```
pbmm2 align ref.fa movie1.subreads.bam ref.movie1.bam --sort --median-filter --sample sample1
```

CCS reads BAM input:

```
pbmm2 align ref.fa movie1.ccs.bam ref.movie1.bam --sort --preset CCS --sample sample1
```

CCS reads FASTQ input:

```
pbmm2 align ref.fa movie1.Q20.fastq ref.movie1.bam --sort --preset CCS --sample sample1 --rg '@RG\tID:movie1'
```

2. Discover the signatures of structural variation, per movie or per sample:

```
pbsv discover ref.movie1.bam ref.sample1.svsig.gz
pbsv discover ref.movie2.bam ref.sample2.svsig.gz
```

3. Call structural variants and assign genotypes (all samples); for CCS analysis input append --ccs:

```
pbsv call ref.fa ref.sample1.svsig.gz ref.sample2.svsig.gz
ref.var.vcf
```

Launching a multi-sample pbsv analysis - requirements

1. Merge multiple Bio Sample SMRT Cells to one Data Set with the Bio Samples specified.
 - Each SMRT Cell must have exactly **one** Bio Sample name - multiple Bio Sample names **cannot** be assigned to one SMRT Cell.
 - **Multiple** SMRT Cells can have the **same** Bio Sample name.
 - **All** of the inputs need to already have the appropriate Bio Sample records in their `CollectionMetadata`. If they don't, they are treated as a **single** sample.
2. Create a ReferenceSet from a FASTA file.
 - The ReferenceSet is often already generated and registered in SMRT Link.
 - If the ReferenceSet doesn't exist, use the `dataset create` command to create one:

```
dataset create --type ReferenceSet --name reference_name reference.fasta
```

Launching a multi-sample analysis

```
# Set subreads and ref FASTA
sample1=sample1.subreadset.xml sample2=sample2.subreadset.xml
ref=reference.fasta

pbmm2 align ${ref} ${sample1} sample1.bam --sort --median-filter --sample Sample1
pbmm2 align ${ref} ${sample2} sample2.bam --sort --median-filter --sample Sample2
samtools index sample1.bam
samtools index sample2.bam
pbindex sample1.bam
pbindex sample2.bam
pbsv discover sample1.bam sample1.svsig.gz
pbsv discover sample2.bam sample2.svsig.gz
pbsv call ${ref} sample1.svsig.gz sample2.svsig.gz out.vcf
```

out.vcf: A pbsv VCF output file, where columns starting from column 10 represent structural variants of Sample 1 and Sample 2:

```
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT Sample1 Sample2
chr01 222737 pbsv.INS.1 T TTGGTGTGTTGTTGTTTGTGTTT . PASS
SVTYPE=INS;END=222737;SVLEN=21;SVANN=TANDEM GT:AD:DP 0/1:6,4:10 0/1:6,5:11
```

pbvalidate The `pbvalidate` tool validates that files produced by PacBio software are compliant with PacBio's own internal specifications.

Input files

`pbvalidate` supports the following input formats:

- BAM
- FASTA
- Data Set XML

See [here](#) for further information about each format's requirements.

Usage

```
pbvalidate [-h] [--version] [--log-file LOG_FILE]
           [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug | --quiet | -v]
           [-c] [--quick] [--max MAX_ERRORS]
           [--max-records MAX_RECORDS]
           [--type {BAM,Fasta,AlignmentSet,ConsensusSet,ConsensusAlignmentSet,
SubreadSet,BarcodeSet,ContigSet,ReferenceSet,HdfSubreadSet}]
           [--index] [--strict] [-x XUNIT_OUT] [--unaligned]
           [--unmapped] [--aligned] [--mapped]
           [--contents {SUBREAD,CCS}] [--reference REFERENCE]
           file
```

Required	Description
file	Input BAM, FASTA, or Data Set XML file to validate.

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. Default (None) will write to stdout.
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = CRITICAL)
--debug=False	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
--verbose, -v	Sets the verbosity level. (Default = None)
--quick	Limits validation to the first 100 records (plus file header); equivalent to --max-records=100. (Default = False)
--max MAX_ERRORS	Exits after MAX_ERRORS were recorded. (Default = None; checks the entire file.)
--max-records MAX_RECORDS	Exits after MAX_RECORDS were inspected. (Default = None; checks the entire file.)
--type	Uses the specified file type instead of guessing. [BAM, Fasta, AlignmentSet, ConsensusSet, ConsensusAlignmentSet, SubreadSet, BarcodeSet, ContigSet, ReferenceSet, HdfSubreadSet] (Default = None)

Options	Description
--index	Requires index files: .fai or .pbi. (Default = False)
--strict	Turns on additional validation, primarily for Data Set XML. (Default = False)

BAM options	Description
--unaligned	Specifies that the file should contain only unmapped alignments. (Default = None, no requirement.)
--unmapped	Alias for --unaligned. (Default = None)
--aligned	Specifies that the file should contain only mapped alignments. (Default = None, no requirement.)
--mapped	Alias for --aligned. (Default = None)
--contents	Enforces the read type: [SUBREAD, CCS] (Default = None)
--reference REFERENCE	Specifies the path to an optional reference FASTA file, used for additional validation of mapped BAM records. (Default = None)

Examples

To validate a BAM file:

```
pbvalidate in.subreads.bam
```

To validate a FASTA file:

```
pbvalidate in.fasta
```

To validate a Data Set XML file:

```
pbvalidate in.subreadset.xml
```

To validate a BAM file and its index file (.pbi):

```
pbvalidate --index in.subreads.bam
```

To validate a BAM file and exit after 10 errors are detected:

```
pbvalidate --max 10 in.subreads.bam
```

To validate up to 100 records in a BAM file:

```
pbvalidate --max-records 100 in.subreads.bam
```

To validate up to 100 records in a BAM file
(equivalent to --max-records=100):

```
pbvalidate --quick in.subreads.bam
```

To validate a BAM file, using a specified log level:

```
pbvalidate --log-level=INFO in.subreads.bam
```

To validate a BAM file and write log messages to a file rather than to stdout:

```
pbvalidate --log-file validation_results.log in.subreads.bam
```

primrose The `primrose` tool analyzes the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.

`primrose` uses a convolution neural network (CNN) to predict the methylation state (5mC) of each CpG in a HiFi read. Methylation is assumed to be symmetric between strands with output reported in the forward direction with respect to the HiFi read sequence. The output uses the `Mm` and `Ml` tags as defined in the SAM Format Optional Fields Specification. (See [here](#) for details.)

Algorithm

The CNN is trained using invitro modified controls of methylated and unmethylated human DNA. The unmethylated control comprises a human shotgun library that has undergone whole genome amplification (WGA), a process that includes PCR, and therefore removes all modifications. The methylated control is generated by subjecting a human WGS library to invitro methylation using a CpG methyltransferase enzyme (*M. SssI*). Kinetic data, pulse width and inter-pulse distance, over a 16 base pair window for both the forward and reverse strand, is used as input to the CNN. The output of the CNN is a probability scale measure of whether the CpG is symmetrically 5mC-modified.

Usage

```
primrose [options] <HiFi INPUT> <HiFi OUTPUT>
```

Required	Description
HiFi INPUT	Input BAM file or ConsensusReadSet XML file.
HiFi OUPUT	Output BAM file or ConsensusReadSet XML file.

Options	Description
<code>--keep-kinetics</code>	Specifies that the kinetics tracks (IPD and PulseWidth records) <code>fi</code> , <code>fp</code> , <code>fn</code> , <code>ri</code> , <code>rp</code> and <code>rn</code> are included in the output BAM file.
<code>--min-passes</code>	Specifies the minimum number of passes. (Default = 3)
<code>--model</code>	Specifies a path to a trained TensorFlow model directory, or to an exported ONNX model file.
<code>-h</code> , <code>--help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-j</code> , <code>--num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stderr</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARNING, , FATAL.] (Default = WARNING)

Example

```
primrose --log-level INFO movie.hifi.bam movie.hifi_5mc.bam
```

runqc-reports The `runqc-reports` tool generates up to five different Run QC reports, depending on Data Set type: Raw Data, Adapters, Loading, Control, and CCS reads. Generating a complete set of reports requires the presence of an `sts.xml` resource in the Data Set, but either the CCS Analysis report (or a fallback Subreads report) will **always** be generated. All report JSON and plot PNG files are written to the current working directory, unless otherwise specified.

Usage

```
runqc-reports [-h] [--version] [--log-file LOG_FILE]
               [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}]
               [| --debug | --quiet | -v]
               [-o OUTPUT_DIR]
               dataset_xml
```

Required	Description
<code>dataset_xml</code>	Input SubreadSet or ConsensusReadSet XML, which must contain an <code>sts.xml</code> resource for the full Run QC report set to be generated.
<code>-o OUTPUT_DIR</code>	Output report directory. (Default = Current working directory)

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stdout</code> .
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = WARNING)
<code>--debug</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--quiet</code>	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = None)

Examples

```
runqc-reports moviename.subreadset.xml
```

```
runqc-reports moviename.consensusreadset.xml
```

summarize Modifications The `summarizeModifications` tool generates a GFF summary file (`alignment_summary.gff`) from the output of base modification analysis (for example, `ipdSummary`) combined with the coverage summary GFF generated by resequencing pipelines. This is useful for power users running custom workflows.

Usage

```
summarizeModifications [-h] [--version]
                        [--log-file LOG_FILE]
                        [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug
                        | --quiet | -v]
                        modifications alignmentSummary gff_out
```

Input files

- `modifications`: Base Modification GFF file.
- `alignmentSummary`: Alignment Summary GFF file.

Output files

- `gff_out`: Coverage summary for regions (bins) spanning the reference with Base Modification results for each region.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to stdout.
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL] (Default = INFO)
<code>--debug</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--quiet</code>	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = None)

Appendix A - Application entry points and output files

Note: To print information about a specific PacBio workflow, including input files and task options, use the `pbcromwell show-workflow-details` command, which is available for **all** applications. **Example:**

```
pbcromwell show-workflow-details pb_hgap4
pbcromwell show-workflow-details cromwell.workflows.pb_hgap4
```

(The prefix `cromwell.workflows` is optional.)

CCS Analysis **Analysis application name:** `cromwell.workflows.pb_ccs`

Entry point

```
:id: eid_subread
:name: Entry eid_subread
:fileTypeId: PacBio.DataSet.SubreadSet
```

Key output files

File name	Datastore SourceId
FASTQ file	<code>ccs_fastq_out</code>
FASTA file	<code>ccs_fasta_out</code>
<movienam>.hifi.reads.bam file	<code>ccs_bam_out</code>
Consensus Sequences	<code>pb_ccs.ccsxml</code>
CCS Analysis Statistics	<code>pb_ccs.report_ccs</code>
All Reads (BAM)	<code>reads_bam</code>
<movienam>.hifi.reads.fasta	<code>ccs_fasta_out</code>
<movienam>.hifi.reads.fastq	<code>ccs_fastq_out</code>

Demultiplex Barcodes **Analysis application name:** `cromwell.workflows.pb_demux_ccs`

Entry points

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_barcode
:name: Entry eid_barcode
:fileTypeId: PacBio.DataSet.BarcodeSet
```

Key output files

File name	Datastore SourceId
Barcode Report Details	<code>pb_demux_ccs.summary_csv</code>
Demultiplexed Datasets	<code>pb_demux_ccs.demuxed_files_datastore</code>
Unbarcoded Reads	<code>pb_demux_ccs.unbarcoded</code>

Export Reads **Analysis application name:** cromwell.workflows.pb_export_ccs**Entry point**

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet
```

Key output files

File name	Datastore SourceId
<moviename>.hifi_reads.fastq.gz	ccs_fastq_out
<moviename>.hifi_reads.fasta.gz	ccs_fasta_out
<moviename>.hifi_reads.bam.gz	ccs_bam_out

Note: If users select a lower cutoff Phred QV value, the string *hifi* is replaced by the QV value in the file names.

Example: <moviename>.q10.fastq.gz.

Genome Assembly

Analysis application name: cromwell.workflows.pb_assembly_hifi

Entry point

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet
```

Key output files

File name	Datastore SourceId
Final Polished Assembly, Primary Contigs	pb_assembly_hifi.final_primary_contigs_fasta
Final Polished Assembly, Haplotigs	pb_assembly_hifi.final_haplotigs_fasta
List of Circular Contigs	pb_assembly_hifi.circular_contigs
Summary Report	pb_assembly_hifi.report_polished_assembly

HiFi Mapping **Analysis application name:** cromwell.workflows.pb_ccs_subreads**Entry points**

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_ref_dataset
:name: Entry eid_ref_dataset
:fileTypeId: PacBio.DataSet.ReferenceSet
```

Key output files

File name	Datastore SourceId
Mapped reads	pb_align_ccs.mapped
Coverage summary	pb_align_ccs.coverage_gff

HiFiViral SARS-CoV-2 Analysis**Analysis application name:** cromwell.workflows.pb_sars_cov2_kit**Entry points**

```

: id: eid_ccs (HiFi reads, demultiplexed as separate BAM files)
: name: Entry_eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_ref_dataset_2 (Reference Genome)
: name: Entry_eid_ref_dataset
: fileId: PacBio.DataSet.ReferenceSet

```

Key output files

File name	Datastore SourceId
All Samples, Probe Counts TSV	pb_sars_cov2_kit.probe_counts_zip
All Samples, Raw Variant Calls VCF	pb_sars_cov2_kit.raw_vcf_zip
All Samples, Variant Call VCF	pb_sars_cov2_kit.vcf_zip
All Samples, Variant Calls CSV	pb_sars_cov2_kit.variants_csv
All Samples, Consensus Sequence FASTA	pb_sars_cov2_kit.fasta_zip
All Samples, Consensus Sequence By Fragments FASTA	pb_sars_cov2_kit.frag_fasta_zip
All Samples, Aligned Reads BAM	pb_sars_cov2_kit.mapped_zip
All Samples, Consensus Sequence Aligned BAM	pb_sars_cov2_kit.aligned_frag_zip
All Samples, Trimmed HiFi reads FASTQ	pb_sars_cov2_kit.trimmed_zip
Failed Sample Info	pb_sars_cov2_kit.sample_failures_csv
Failed Sample Analysis Logs	pb_sars_cov2_kit.errors_zip
Demultiplex Summaries	pb_sars_cov2_kit.lima_summary_zip
Sample Summary Table CSV	pb_sars_cov2_kit.summary_csv
All Samples, Genome Coverage Plots	pb_sars_cov2_kit.coverage_png_zip

Iso-Seq Analysis**Analysis application name:** cromwell.workflows.pb_isoseq3_ccsonly**Entry points**

```

: id: eid_ccs
: name: Reads
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_barcode
: name: Primers
: fileId: PacBio.DataSet.BarcodeSet

: id: eid_ref_dataset
: name: Reference (Optional)
: fileId: PacBio.DataSet.ReferenceSet

```


Key output files

File name	Datastore SourceId
Collapsed Filtered Isoforms FASTQ	pb_isoseq3_ccsonly.collapse_fastq
Collapsed Filtered Isoforms GFF	pb_isoseq3_ccsonly.collapse_gff
Group TXT	pb_isoseq3_ccsonly.collapse_group
Abundance TXT	pb_isoseq3_ccsonly.collapse_abundance
Read Stat TXT	pb_isoseq3_ccsonly.collapse_readstat
Collapsed Isoforms Abundance TXT (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.collapse_abundance Separate clusters, one per barcoded sample.
Collapsed Isoforms Abundance TXT (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.collapse_abundance Pooled clusters, one per barcoded sample.
High-Quality Transcripts	pb_isoseq3_ccsonly.hq_fastq
Low-Quality Transcripts	pb_isoseq3_ccsonly.lq_fastq
High-Quality Transcripts Counts (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.barcode_overview_report Separate clusters, one per barcoded sample.
High-Quality Transcripts Counts (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.barcode_overview_report Pooled clusters, one per barcoded sample.
CCS reads FASTQ	pb_isoseq3_ccsonly.ccs_fastq_zip
Full-length CCS reads	pb_isoseq3._ccsonly.flnc_bam
Polished Report	pb_isoseq3._ccsonly.polish_report_csv
Cluster Report	pb_isoseq3._ccsonly.report_isoseq

**Mark PCR
Duplicates****Analysis application name:** cromwell.workflows.pb_mark_duplicates**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

```

Key output files

File name	Datastore SourceId
PCR Duplicates	pb_mark_duplicates.duplicates
Deduplicated reads	pb_mark_duplicates.deduplicated In the SMRT Link UI, this displays as <ORIGINAL_DATASET_NAME> (deduplicated).

**Microbial
Genome
Analysis****Analysis application name:** cromwell.workflows.pb_microbial_analysis**Entry point**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusreadSet

```

Key output files

File name	Datastore SourceId
Final Polished Assembly	pb_microbial_analysis.assembly_fasta
Final Polished Assembly Index	pb_microbial_analysis.assembly_fasta.fai
Final Polished Assembly for NCBI	pb_microbial_analysis.ncbi_fasta
Mapped BAM	pb_microbial_analysis.mapped
Target sequences used to produce the Mapped BAM	pb_microbial_analysis.mapped_target_fasta
Target sequence Index used to produce the Mapped BAM	pb_microbial_analysis.mapped_target_fasta_fai
List of Circular Contigs	pb_microbial_analysis.circular_list
Coverage Summary	pb_microbial_analysis.coverage_gff
Coverage Report	pb_microbial_analysis.report_coverage
Mapping Statistics Report	pb_microbial_analysis.report_mapping_stats
Mapped BAM Datastore	pb_microbial_analysis.mapped_bam_datastore
Polished Assembly Report	pb_microbial_analysis.report_polished_assembly
Full Kinetics Summary	pb_microbial_analysis.basemods_gff
IPD Ratios	pb_microbial_analysis.basemods_csv
Motifs and Modifications	pb_microbial_analysis.motifs_csv
Motifs and Modifications	pb_microbial_analysis.motifs_gff

**Minor Variants
Analysis****Analysis application name:** cromwell.workflows.pb_mv_ccs**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_ref_dataset
: name: Entry eid_ref_dataset
: fileId: PacBio.DataSet.ReferenceSet

```

Key output files

File name	Datastore SourceId
Minor Variants HTML Reports	pb_mv_ccs.juliet_html
Per-Variant Table	pb_mv_ccs.report_csv
Alignments	pb_mv_ccs.mapped

**Structural
Variant Calling****Analysis application name:** cromwell.workflows.pb_sv_ccs**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_ref_dataset
: name: Entry eid_ref_dataset
: fileId: PacBio.DataSet.ReferenceSet

```

Key output files

File name	Datastore SourceId
Structural Variants	pb_sv_ccs.variants
Aligned reads (Bio Sample Name)	pb_sv_ccs.alignments_by_sample_datastore

**Trim Ultra-Low
Adapters****Analysis application name:** cromwell.workflows.pb_trim_adapters**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_barcode
: name: Entry eid_barcode
: fileId: PacBio.DataSet.BarcodeSet

```

Note: The barcodes need to be a single primer sequence.**Key output files**

File name	Datastore SourceId
Reads Missing Adapters	pb_trim_adapters.unbarcoded
PCR Adapter Data CSV	pb_trim_adapters.summary_csv
Trimmed reads	pb_trim_adapters.trimmed In the SMRT Link UI, this displays as <ORIGINAL_DATASET_NAME> (trimmed).

**5mC CpG
Detection****Analysis application name:** cromwell.workflows.pb_detect_methyl**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_barcode
: name: Entry eid_barcode
: fileId: PacBio.DataSet.BarcodeSet

```

Key output files

File name	Datastore SourceId
HiFi reads with 5mC Calls	pb_detect_methyl.ccsbam

Appendix B - Third party command-line tools

Following is information on the third-party command-line tools included in the `smrtcnds/bin` subdirectory.

- | | |
|-------------------------------|--|
| bamtools | <ul style="list-style-type: none">• A C++ API and toolkit for reading, writing, and manipulating BAM files.• See https://sourceforge.net/projects/bamtools/ for details. |
| cromwell | <ul style="list-style-type: none">• Scientific workflow engine used to power SMRT Link.• See https://cromwell.readthedocs.io/en/stable/ for details. |
| ipython | <ul style="list-style-type: none">• An interactive shell for using the PacBio API.• See https://ipython.org/ for details. |
| purge_dups | <ul style="list-style-type: none">• Removes haplotigs and contig overlaps in a <i>de novo</i> assembly based on read depth.• See https://github.com/dfguan/purge_dups for details. |
| python | <ul style="list-style-type: none">• An object-oriented programming language.• See https://www.python.org/ for details. |
| samtools,
BCFtools | <ul style="list-style-type: none">• A set of programs for interacting with high-throughput sequencing data in SAM/BAM/CRAM/VCF/BCF2 formats.• See http://www.htslib.org/ for details. |

Appendix C - Microbial Genome Analysis advanced options (HiFi reads only)

Use this application to generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb from HiFi (CCS) reads.

The workflow also performs base modification analysis of the assembled genome.

The Microbial Genome Analysis application:

- Includes chromosomal- and plasmid-level *de novo* genome assembly, circularization, polishing, and rotation of the origin of replication for each circular contig.
- Performs base modification detection (m4C and m6A).
- Facilitates assembly of larger genomes (yeast) as well.
- Accepts HiFi read data (BAM format) as input.

The workflow consists of two assembly stages: **chromosomal** and **plasmid**.

Chromosomal stage: Intended for contig assembly of large sequences. This stage uses more stringent filtering (using advanced options) to produce contiguous assemblies of complex regions, but it may miss small sequences in the input sample (such as plasmids.)

Plasmid stage: Intended for a fine-grained assembly. This stage assembles **only** the unmapped and poorly mapped reads. It also relaxes the overlapping parameters, using advanced options.

Both stages use an automated random subsampling process to reduce the input Data Set for assembly (by default to 100x). Note that the subsampling is **only** applied to the contig construction process, while the alignment and reports stages of the workflow still uses the **full** input Data Set.

Available options for the two assembly stages are **identical**. The only differences are:

1. Chromosomal stage parameters are prefixed with:
`ipa2_advanced_options_chrom`. In the SMRT Link interface for the Microbial Genome Analysis application, this corresponds to the **Advanced Assembly Options for chromosomal stage** field.
2. Plasmid stage parameters are prefixed with:
`ipa2_advanced_options_plasmid`. In the SMRT Link interface for the Microbial Genome Analysis application, this corresponds to the **Advanced Assembly Options for plasmid stage** field.

The same sub-options are available to each stage, but the defaults are very different. The current defaults are:

```
ipa2_advanced_options_chrom =
```

```
"config_block_size = 100; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75;"
```

```
ipa2_advanced_options_plasmid =
```

```
"config_block_size = 100; config_ovl_filter_opt = --max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 500 --gapFilt --minDepth 4 --idt-stage2 98; config_ovl_min_len = 500; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --min-map-len 500 --min-anchor-span 500 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75 --smart-hit-per-target --secondary-min-ovl-frac 0.05; config_layout_opt = --allow-circular;"
```

Options are separated by **semicolons**; within each option, parameters are separated by **spaces**.

Users should not need to modify any of default options. If the defaults are modified, workflow behavior could be very different.

Note: The options available in `ipa2_advanced_options_*` are **exactly** the same as the `config_*` options available for the **Genome Assembly** tool. See [“Genome Assembly parameters input files” on page 43](#) for details.

EXHIBIT K

[← All videos](#)

OCTOBER 28, 2021 | PRESENTATION

Methylation Detection with PacBio HiFi Sequencing

[Twitter](#) [LinkedIn](#) [Facebook](#)

In this talk, Dr. Aaron Wenger, describes how PacBio HiFi reads (15 kb – 25 kb, >99.9% accuracy) provide the most complete view of human genetic variation, including small variants in difficult-to-map regions and structural variants genome-wide. Further, PacBio sequencing simultaneously detects epigenetic modifications without requiring a specialized library preparation step like bisulfite conversion. This ability is commonly used to characterize epigenetic marks in bacterial genomes. Recent improvements in read length and data analysis have extended the ability to include the 5mC methylation that is present at CpG sites in human genomes. Using a deep learning model that integrates sequencing kinetics and base context, the accuracy of 5mC detection in humans for individual HiFi reads is around 80%. Combining multiple reads, the concordance to EMseq and bisulfite sequencing reaches around 90%. The single-molecule resolution of methylation, together with phasing from accurate long reads, allows the detection of allele-specific methylation patterns such as parental imprinting. This ability to detect bases and modifications allows HiFi sequencing to provide the most complete genome and epigenomes with a single technology and library preparation.



TALK WITH AN EXPERT

If you have a question, need to check the status of an order, or are interested in purchasing an instrument, we're here to help.

Area of interest *

Questions *

[Connect with a PacBio scientist](#)By registering on this web page, you are consenting and agreeing to collection and use of that information by PacBio in accordance with our [privacy policy](#).

Our revolutionary sequencing technologies combine the completeness of long reads with the accuracy of short reads to provide the most comprehensive view of genomes, transcriptomes, and epigenomes.

Connect with us



Email *

[Join our mailing list](#)**PRODUCTS**

HiFi sequencing
Sequel IIe system
Sequencing methods
Consumables
Data analysis
Purchase
Documentation
Software downloads

FOCUS AREAS

Human genomics
Plant + animal
Infectious disease
Microbial genomics
Cancer research
Gene therapy

ENGAGE

Blog
Webinars
Events
Case studies
Sequencing 101
Access datasets
Resources
Become a service provider
Find a service provider
Find a distributor

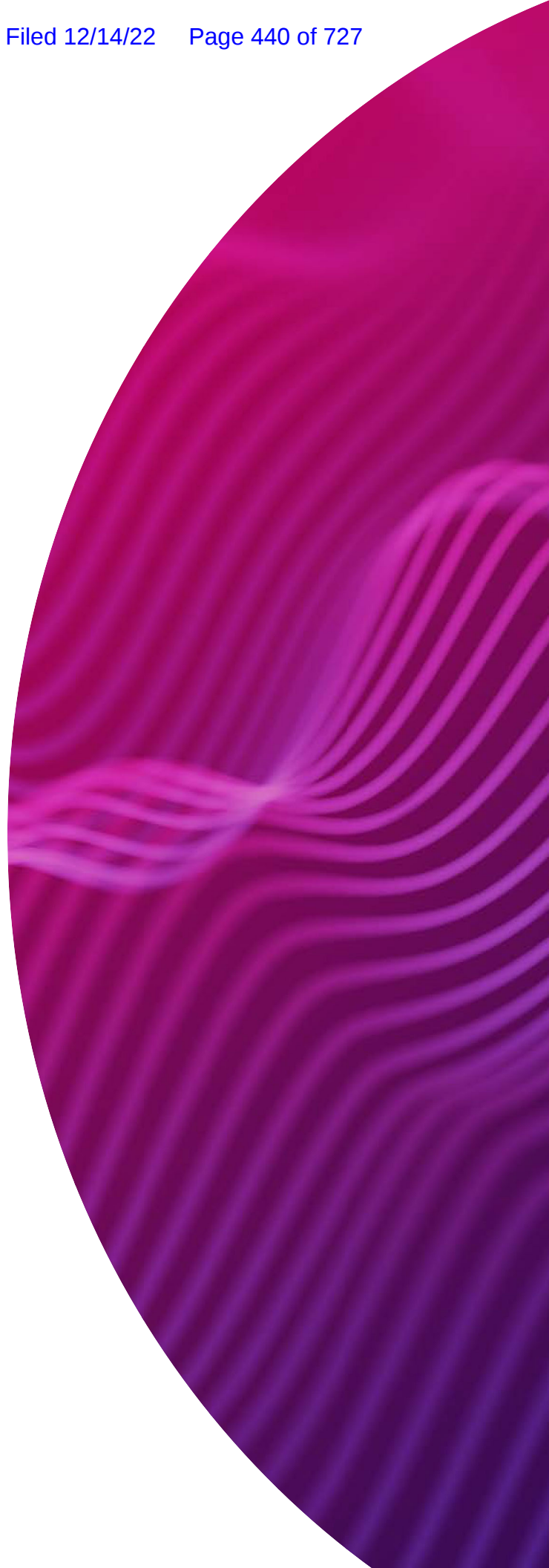
COMPANY

About us
Contact us
Leadership
Press releases
Investors
Sustainability
Careers
Support
Customer login

EXHIBIT L



SMRT[®] Tools reference guide (v11.1)



Research use only. Not for use in diagnostic procedures.

P/N 102-413-200 Version 01 (November 2022)

© 2022, PacBio. All rights reserved.

Information in this document is subject to change without notice. PacBio assumes no responsibility for any errors or omissions in this document.

Certain notices, terms, conditions and/or use restrictions may pertain to your use of PacBio products and/or third party products. Refer to the applicable PacBio terms and conditions of sale and to the applicable license terms at <http://www.pacificbiosciences.com/licenses.html>.

Trademarks:

Pacific Biosciences, the PacBio logo, PacBio, Circulomics, Omnione, SMRT, SMRTbell, Iso-Seq, Sequel, Nanobind, and SBB are trademarks of Pacific Biosciences of California Inc. (PacBio). All other trademarks are the sole property of their respective owners.

See <https://github.com/broadinstitute/cromwell/blob/develop/LICENSE.txt> for Cromwell redistribution information.

PacBio

1305 O'Brien Drive

Menlo Park, CA 94025

www.pacb.com



SMRT® Tools reference guide (v11.1)

Introduction

This document describes the command-line tools included with SMRT® Link v11.1. These tools are for use by bioinformaticians working with secondary analysis results.

- The command-line tools are located in the `$SMRT_ROOT/smrtlink/smrtcmds/bin` subdirectory.

Installation

The command-line tools are installed as an integral component of the SMRT Link software. For installation details, see **SMRT Link software installation guide (v11.1)**.

- **Note:** SMRT Link v11.1 is for use with Sequel® II systems and Sequel IIe systems **only**.
- To install **only** the command-line tools, use the `--smrttools-only` option with the installation command, whether for a new installation or an upgrade. **Examples:**

```
smrtlink-*.run --rootdir smrtlink --smrttools-only
smrtlink-*.run --rootdir smrtlink --smrttools-only --upgrade
```

Supported chemistry

SMRT Link v11.1 supports all chemistry versions for Sequel II systems and Sequel IIe systems.

PacBio command-line tools

Following is information on the PacBio-supplied command-line tools included in the installation. For information on third-party tools installed, see [“Appendix B - Third party command-line tools” on page 130](#).

Tool	Description
bam2fasta/ bam2fastq	Converts PacBio® BAM files into gzipped FASTA and FASTQ files. See “bam2fasta/bam2fastq” on page 4 .
bamsieve	Generates a subset of a BAM or PacBio Data Set file based on either a list of hole numbers, or a percentage of reads to be randomly selected. See “bamsieve” on page 5 .
ccs	Calculates consensus sequences from multiple “passes” around a circularized single DNA molecule (SMRTbell® template). See “ccs” on page 8 .
dataset	Creates, opens, manipulates and writes Data Set XML files. See “dataset” on page 17 .

Tool	Description
Demultiplex Barcodes	Identifies barcode sequences in PacBio single-molecule sequencing data. See “Demultiplex Barcodes” on page 24.
export-datasets	Takes one or more PacBio dataset XML files and packages all contents into a single ZIP archive. See “export-datasets” on page 36.
export-job	Takes one SMRT Link analysis job and packages all contents into a single ZIP archive. See “export-job” on page 37.
gcpp	Variant-calling tool which provides several variant-calling algorithms for PacBio sequencing data. See “gcpp” on page 39.
Genome Assembly	Generates <i>de novo</i> assemblies using HiFi reads. See “Genome Assembly” on page 42.
HiFiViral SARS-CoV-2 Analysis	Analyzes multiplexed viral surveillance samples for SARS-CoV-2. See “HiFiViral SARS-CoV-2 Analysis” on page 50.
ipdSummary	Detects DNA base-modifications from kinetic signatures. See “ipdSummary” on page 53.
isoseq3	Characterizes full-length transcripts and generates full-length transcript isoforms, eliminating the need for computational reconstruction. See “isoseq3” on page 57.
juliet	A general-purpose minor variant caller that identifies and phases minor single nucleotide substitution variants in complex populations. See “juliet” on page 66.
Microbial Genome Analysis	Generates <i>de novo</i> assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb and performs base modification detection, using HiFi reads. See “Microbial Genome Analysis” on page 74.
motifMaker	Identifies motifs associated with DNA modifications in prokaryotic genomes. See “motifMaker” on page 78.
pbcromwell	PacBio’s wrapper for the <code>cromwell</code> scientific workflow engine used to power SMRT Link. For details on how to use <code>pbcromwell</code> to run workflows, see “pbcromwell” on page 80.
pbindex	Creates an index file that enables random access to PacBio-specific data in BAM files. See “pbindex” on page 85.
pbmarkdup	Marks or removes duplicates reads from CCS reads. See “pbmarkdup” on page 86.
pbmm2	Aligns PacBio reads to reference sequences; a SMRT wrapper for <code>minimap2</code> . See “pbmm2” on page 89.
pbservice	Performs a variety of useful tasks within SMRT Link. See “pbservice” on page 96.
pbsv	Structural variant caller for PacBio reads. See “pbsv” on page 101.
pbvalidate	Validates that files produced by PacBio software are compliant with PacBio’s own internal specifications. See “pbvalidate” on page 105.
pigeon	Classifies and filters full-length transcript isoforms into categories against a reference annotation. See “pigeon” on page 108.
primrose	Performs motif-calling to detect 5mC CpG sites in HiFi BAM files. See “primrose” on page 114.
runqc-reports	Generates Run QC reports. See “runqc-reports” on page 115.
skera	Splits arrayed HiFi reads at adapter positions, generating read-segments. See “skera” on page 116.

Tool	Description
summarizeModifications	Generates a GFF summary file from the output of base modification analysis combined with the coverage summary GFF generated by resequencing pipelines. See “summarize Modifications” on page 120 .

**bam2fasta/
bam2fastq**

The `bam2fasta` and `bam2fastq` tools convert PacBio BAM or Data Set files into gzipped FASTA and FASTQ files, including demultiplexing of barcoded data.

Usage

Both tools have an identical interface and take BAM and/or Data Set files as input.

```
bam2fasta [options] <input>
bam2fastq [options] <input>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-o, --output</code>	Specifies the prefix of the output file names. <code>-</code> implies streaming. Note: Streaming is not supported with compression or with the <code>split_barcodes</code> option.
<code>-c</code>	Specifies the Gzip compression level. Values are <code>[1, 2, 3, 4, 5, 6, 7, 8, 9]</code> . (Default = 1)
<code>-u</code>	Specifies that the output FASTA/FASTQ files are not compressed. <code>.gz</code> is not added to the output file names, and <code>-c</code> settings are ignored.
<code>--split-barcodes</code>	Splits the output into multiple FASTA/FASTQ files, by barcode pairs. Note: The <code>bam2fasta/bam2fastq</code> tools inspect the <code>bc</code> tag in the BAM file to determine the 0-based barcode indices from their respective positions in the barcode FASTA file.
<code>-p, --seqid-prefix</code>	Specifies the prefix for the sequence IDs used in the FAST/FASTQ file headers.

Examples

```
bam2fasta -o projectName m54008_160330_053509.subreads.bam
```

```
bam2fastq -o myEcoliRuns m54008_160330_053509.subreads.bam
m54008_160331_235636.subreads.bam
```

```
bam2fasta -o myHumanGenome m54012_160401_000001.subreadset.xml
```

Input files

- One or more `*.bam` files
- `*.subreadset.xml` file (Data Set file)

Output files

- `*.fasta.gz`
- `*.fastq.gz`

bamsieve The `bamsieve` tool creates a subset of a BAM or PacBio Data Set file based on either a list of hole numbers to include or exclude, or a percentage of reads to be randomly selected, while keeping all subreads within a read together. Although `bamsieve` is BAM-centric, it has some support for dataset XML and will propagate metadata, as well as scraps BAM files in the special case of SubreadSets. `bamsieve` is useful for generating minimal test Data Sets containing a handful of reads.

`bamsieve` operates in two modes: **list** mode where the ZMWs to keep or discard are explicitly specified, or **percentage/count** mode, where a fraction of the ZMWs is randomly selected.

ZMWs may be listed in one of several ways:

- As a comma-separated list on the command line.
- As a flat text file, one ZMW per line.
- As another PacBio BAM or Data Set of any type.

Usage

```
bamsieve [-h] [--version] [--log-file LOG_FILE]
          [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}] | --debug | --quiet
          | -v]
          [--show-zmws] [--include INCLUDE LIST] [--exclude EXCLUDE LIST]
          [--percentage PERCENTAGE] [-n COUNT] [-s SEED]
          [--ignore-metadata] [--barcodes]
          input_bam [output_bam]
```

Required	Description
input_bam	The name of the input BAM file or Data Set from which reads will be read.
output_bam	The name of the output BAM file or Data Set where filtered reads will be written to. (Default = None)

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. (Default = None, writes to stdout.)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL]. (Default = WARNING)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
-v, --verbose	Sets the verbosity level. (Default = NONE)
--show-zmws	Prints a list of ZMWs and exits. (Default = False)
--include INCLUDE LIST	Specifies the ZMWs to include in the output. This can be a comma-separated list of ZMWs, or a file containing a list of ZMWs (one hole number per line), or a BAM/ Data Set file. (Default = NONE)

Options	Description
<code>--exclude EXCLUDE LIST</code>	Specifies the ZMWs to exclude from the output. This can be a comma-separated list of ZMWs, or a file containing a list of ZMWs (one hole number per line), or a BAM/Data Set file that specifies ZMWs. (Default = NONE)
<code>--percentage PERCENTAGE</code>	Specifies a percentage of a SMRT® Cell to recover (Range = 1-100) rather than a specific list of reads. (Default = NONE)
<code>-n COUNT, --count COUNT</code>	Specifies a specific number of ZMWs picked at random to recover. (Default = NONE)
<code>-s SEED, --seed SEED</code>	Specifies a random seed for selecting a percentage of reads. (Default = NONE)
<code>--ignore-metadata</code>	Discard the input Data Set metadata. (Default = False)
<code>--barcodes</code>	Specifies that the include/exclude list contains barcode indices instead of ZMW numbers. (Default = False)

Examples

Pulling out two ZMWs from a BAM file:

```
bamsieve --include 111111,222222 full.subreads.bam sample.subreads.bam
```

Pulling out two ZMWs from a Data Set file:

```
bamsieve --include 111111,222222 full.subreadset.xml sample.subreadset.xml
```

Using a text list:

```
bamsieve --include zmw.txt full.subreads.bam sample.subreads.bam
```

Using another BAM or Data Set as a list:

```
bamsieve --include mapped.alignmentset.xml full.subreads.bam mappable.subreads.bam
```

Generating a list of ZMWs from a Data Set:

```
bamsieve --show-zmws mapped.alignmentset.xml > mapped_zmws.txt
```

Anonymizing a Data Set:

```
bamsieve --include zmw.txt --ignore-metadata --anonymize full.subreadset.xml  
anonymous_sample.subreadset.xml
```

Removing a read:

```
bamsieve --exclude 111111 full.subreadset.xml filtered.subreadset.xml
```

Selecting 0.1% of reads:

```
bamsieve --percentage 0.1 full.subreads.bam random_sample.subreads.bam
```

Selecting a different 0.1% of reads:

```
bamsieve --percentage 0.1 --seed 98765 full.subreads.bam random_sample.subreads.bam
```

Selecting just two ZMWs/reads at random:

```
bamsieve --count 2 full.subreads.bam two_reads.subreads.bam
```

Selecting by barcode:

```
bamsieve --barcodes --include 4,7 full.subreads.bam two_barcodes.subreads.bam
```

Generating a tiny BAM file that contains only mappable reads:

```
bamsieve --include mapped.subreads.bam full.subreads.bam mappable.subreads.bam  
bamsieve --count 4 mappable.subreads.bam tiny.subreads.bam
```

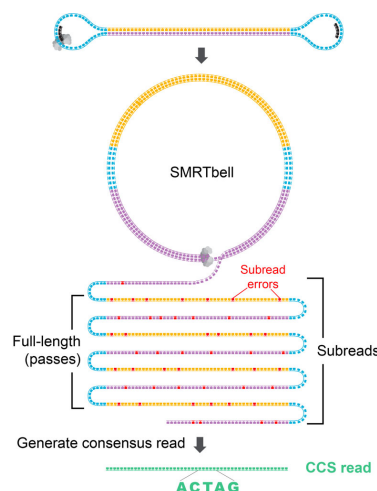
Splitting a Data Set into two halves:

```
bamsieve --percentage 50 full.subreadset.xml split.1of2.subreadset.xml  
bamsieve --exclude split.1of2.subreadset.xml full.subreadset.xml  
split.2of2.subreadset.xml
```

Extracting Unmapped Reads:

```
bamsieve --exclude mapped.alignmentset.xml movie.subreadset.xml unmapped.subreadset.xml
```

ccs Circular Consensus Sequencing (CCS) Analysis computes consensus sequences from multiple “passes” around a circularized single DNA molecule (SMRTbell® template). CCS analysis uses the Arrow framework to achieve optimal consensus results given the number of passes available.



CCS analysis workflow

1. Initial filtering

- Filter ZMWs: Remove ZMWs with signal-to-noise ratio (SNR) below `--min-snr`.
- Filter subreads: Remove subreads with lengths <50% or >200% of the median subread length. Stop if the number of full-length subreads is fewer than `--min-passes`.

2. Generate draft

- The polish stage iteratively improves upon a candidate template sequence. Because polishing is very compute-intensive, it is desirable to start with a template that is as close as possible to the true sequence of the molecule to reduce the number of iterations until convergence. The `ccs` software does **not** pick a full-length subread as the initial template to be polished, but instead generates an approximate draft consensus sequence using our improved implementation of the `sparc` graph consensus algorithm. This algorithm depends on a subread-to-backbone alignment that is generated by the `pancake` mapper developed by PacBio, using `edlib` as the core aligner. Typically, subreads have accuracy of around 90% and the draft consensus has a higher accuracy, but is still below 99%.
- Stop if the draft length is shorter than `--min-length` and longer than `--max-length`.

3. Alignment

- Align subreads to the draft consensus using `pancake` with KSW2 for downstream windowing and filtering.

4. Windowing

- Divide the subread-to-draft alignment into overlapping windows with a target size of 22 bp with ± 2 bp overlap. Avoid breaking windows at simple repeats (homopolymers to 4-mer repeats) to reduce edge cases at window borders. Windowing reduces the algorithm run time from quadratic to linear in the insert size.

5. Single-strand artifacts

- Identify heteroduplexes, where one strand of the SMRTbell differs significantly from the reverse complement of the other strand. Subread orientation is inferred from the alignment. Small heteroduplexes, such as single base `A` paired with a matching `G`, are retained and the ambiguity is reflected in base quality. Molecules with a single difference longer than 20 bp between the strands are removed and recorded as heteroduplexes in the `<outputPrefix>.ccs_report.txt` file.

6. Trim large insertions

- Insertions in the subreads relative to the draft that are longer than `--max-insertion-size`, default 30 bp, are trimmed since they typically represented spurious sequencing activity.

7. Filter candidates

- For each window, a heuristic is used to find those positions that likely need polishing. In addition, homopolymers are always polished. Skipping unambiguous positions makes the polishing at least twice as fast.

8. Polishing

- The core polishing uses the `arrow` algorithm, a left-right Hidden Markov-Model (HMM) that models the enzymatic and photophysical events in SMRT sequencing. Emission and transition parameters are estimated by a dinucleotide template context. Transition parameters form a homogeneous Markov chain. The transition parameters do **not** depend on the position within the template, only the pulse width of a base call, the dinucleotide context of the template, and the SNR of the ZMW. `Arrow` computes the log-likelihood that the subread originates from the template, marginalizing over all possible alignments of the subread to the template. For every position in the template that is a candidate for polishing, `arrow` tests if the log-likelihood is improved by substituting one of the other three nucleotides, inserting one of the four nucleotides after the position, or deleting the position itself. Once `arrow` does not find any further beneficial mutations to the template in an iteration, it stops.

9. QV calculation

- The log-likelihood ratio between the most likely template sequence and all of its mutated counterparts is used to calculate a quality for each base in the final consensus. The average of the per-base qualities is the read accuracy, `rq`.

10. Final steps

- The per-window consensus template sequences and base qualities are concatenated and overhangs (overlaps between adjacent windows) are trimmed. If the predicted read accuracy is at least `--min-rq`, then the consensus read is written to the output.

Input files

- One `.subreads.bam` file containing the subreads for each SMRTbell® template sequenced.

Output files

- A BAM file with one entry for each consensus sequence derived from a productive ZMW. BAM is a general file format for storing sequence data, which is described fully by the SAM/BAM working group. The CCS analysis output format is a version of this general format, where the consensus sequence is represented by the "Query Sequence". Several tags were added to provide additional meta information. An example BAM entry for a consensus as seen by `samtools` is shown below.

```
m64009_201008_223950/1/ccs      4      *      0      255      *      *      0      0      ATCGCCTACC
~|~t~R~r~      RG:Z:a773clf2      fi:B:C,26,60,21,41,33,26,63,45,73,33      fn:i:6
fp:B:C,11,18,21,35,8,18,31,8,23,11      np:i:12
ri:B:C,17,37,24,4,70,21,12,44,21,32      rn:i:6      rp:B:C,16,56,17,9,23,19,10,10,23,12
      rq:f:0.999651      sn:B:f,10.999,16.2603,3.964,7.17746      we:i:9816064      ws:i:20
zm:i:1
```

Following are some of the common fields contained in the output BAM file:

Field	Description
Query Name	Movie Name / ZMW # /ccs
FLAG	Required by the format but meaningless in this context. Always set to 4 to indicate the read is unmapped.
Reference Name	Required by the format but meaningless in this context. Always set to *.
Mapping Start	Required by the format but meaningless in this context. Always set to 0.
Mapping Quality	Required by the format but meaningless in this context. Always set to 255.
CIGAR	Required by the format but meaningless in this context. Always set to *.
RNEXT	Required by the format but meaningless in this context. Always set to *.
PNEXT	Required by the format but meaningless in this context. Always set to 0.
TLEN	Required by the format but meaningless in this context. Always set to 0.
Consensus Sequence	The consensus sequence generated.
Quality Values	The per-base parametric quality metric. For details see "Interpreting QUAL values" on page 13 .
RG Tag	The read group identifier.
bc Tag	A 2-entry array of upstream-provided barcode calls for this ZMW.
bq Tag	The quality of the barcode call. (Optional : Depends on barcoded inputs.)

Field	Description
np Tag	The number of full passes that went into the subread. (Optional : Depends on barcoded inputs.)
rq Tag	The predicted read quality.
zm Tag	The ZMW hole number.
ma Tag	Bitmask storing whether a SMRTbell adapter is missing on either side of the molecule that produced a CCS read. 0 indicates that neither end has a confirmed missing adapter.
ac Tag	An array containing the counts of detected and missing SMRTbell adapters on either side of the molecule that produced a CCS read: <ul style="list-style-type: none"> • Detected adapters on left/start • Missing adapters on left/start • Detected adapters on right/end • Missing adapter on right/end

Usage

```
ccs [OPTIONS] INPUT OUTPUT
```

Example

```
ccs --all myData.subreads.bam myResult.bam
```

Required	Description
Input File Name	The name of a single subreads.bam or a subreadset.xml file to be processed. (Example = myData.subreads.bam)
Output File Name	The name of the output BAM file; comes after all other options listed. Valid output files are the BAM and the Dataset.xml formats. (Example = myResult.bam)

Options	Description
--version	Prints the version number.
--report-file	Contains a result tally of the outcomes for all ZMWs that were processed. If no file name is given, the report is output to the file <code>ccs_report.txt</code> . In addition to the count of successfully-produced consensus sequences, this file lists how many ZMWs failed various data quality filters (SNR too low, not enough full passes, and so on) and is useful for diagnosing unexpected drops in yield.
--min-snr	Removes data that is likely to contain deletions. SNR is a measure of the strength of signal for all 4 channels (A, C, G, T) used to detect base pair incorporation. This value sets the threshold for minimum required SNR for any of the four channels. Data with SNR < 2.5 is typically considered lower quality. (Default = 2.5)
--min-length	Specifies the minimum length requirement for the minimum length of the draft consensus to be used for further polishing. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates. (Default = 10)
--max-length	Specifies the maximum length requirement for the maximum length of the draft consensus to be used for further polishing. For robust results while avoiding unnecessary computation on unusual data, set to ~20% above the largest expected insert size. (Default = 50000)
--min-passes	Specifies the minimum number of passes for a ZMW to be emitted. This is the number of full passes. Full passes must have an adapter hit before and after the insert sequence and so do not include any partial passes at the start and end of the sequencing reaction. It is computed as the number of passes made across all windows. (Default = 3)

Options	Description
<code>--min-rq</code>	Specifies the minimum predicted accuracy of a read. CCS analysis generates an accuracy prediction for each read, defined as the expected percentage of matches in an alignment of the consensus sequence to the true read. A value of <code>0.99</code> indicates that only reads expected to be 99% accurate are emitted. (Default = <code>0.99</code>)
<code>--num-threads</code>	Specifies how many threads to use while processing. By default, CCS analysis uses as many threads as there are available cores to minimize processing time, but fewer threads can be specified here.
<code>--log-file</code>	The name of a log file to use. If none is given, the logging information is printed to <code>STDERR</code> . (Example: <code>mylog.txt</code>)
<code>--log-level</code>	Specifies verbosity of log data to produce. By setting <code>--logLevel=DEBUG</code> , you can obtain detailed information on what ZMWs were dropped during processing, as well as any errors which may have appeared. (Default = <code>INFO</code>)
<code>--skip-polish</code>	After constructing the draft consensus, do not proceed with the polishing steps. This is significantly faster, but generates less accurate data with no RQ or QUAL values associated with each base.
<code>--by-strand</code>	Separately generates a consensus sequence from the forward and reverse strands. Useful for identifying heteroduplexes formed during sample preparation.
<code>--chunk</code>	Operates on a single chunk. Format <code>i/N</code> , where <code>i</code> in <code>[1,N]</code> . Examples: <code>3/24</code> or <code>9/9</code> .
<code>--max-chunks</code>	Determines the maximum number of chunks, given an input file.
<code>--model-path</code>	Specifies the path to a model file or directory containing model files.
<code>--model-spec</code>	Specifies the name of the chemistry or model to use, overriding the default selection.
<code>--all</code>	<p>Generates one representative sequence per ZMW, irrespective of quality and passes. <code>--min-passes 0 --min-rq 0 --max-length 0</code> are set implicitly and cannot be changed; <code>--all</code> also deactivates the maximum draft length filter. Filtering has to be performed downstream.</p> <p>The <code>ccs --all</code> option changes the workflow as follows:</p> <ol style="list-style-type: none"> 1. There is special behavior for low-pass ZMWs. If a ZMW has fewer than 2 full-length subreads, use the subread of median length as representative consensus, optionally with its kinetic information as forward orientation using <code>--all-kinetics</code>, and do not polish. 2. Only polish ZMWs with at least two full-length subreads mapping back to the draft. Otherwise, set predicted accuracy <code>rq</code> tag to <code>-1</code> to indicate that the predicted accuracy was not calculated, and populate per-base QVs with <code>+</code> (QV 10) the approximate raw accuracy. Kinetic information is not available for unpolished drafts. 3. Instead of using an unpolished draft without kinetic information as a representative consensus sequence, if <code>--subread-fallback</code> is used, fall back to a representative subread with kinetic information. <p>How is <code>--all</code> different from explicitly setting <code>--min-passes 0 --min-rq 0</code>?</p> <ul style="list-style-type: none"> • Setting <code>--min-passes 0 --min-rq 0</code> is a brute-force combination that polishes every ZMW, even those that only have one partial subread, with polishing making no difference. In contrast, <code>--all</code> is a bit smarter and only polishes ZMWs with at least one full-length subread and one additional partial subread.

Options	Description
<code>--hifi-kinetics</code>	<p>Generates averaged kinetic information for polished reads, independently for both strands of the insert. Forward is defined with respect to the orientation represented in SEQ and is considered to be the native orientation. As with other PacBio-specific tags, aligners will not re-orient these fields.</p> <p>Base modifications can be inferred from per-base pulse width (PW) and inter-pulse duration (IPD) kinetics.</p> <p>Minor cases exist where a certain orientation may get filtered out entirely from a ZMW, preventing valid values from being passed for that record. In these cases, empty lists are passed for the respective record/orientation, and number of passes are set to zero.</p> <p>To facilitate the use of HiFi reads with base modifications workflows, we added an executable in pbbam called <code>ccs-kinetics-bystrandify</code> which creates a pseudo <code>--by-strand</code> BAM with corresponding <code>pw</code> and <code>ip</code> tags that imitates a normal, unaligned subreads BAM.</p>
<code>--all-kinetics</code>	Adds kinetic information for all ZMWs, except for unpolished draft consensus.
<code>--subread-fallback</code>	When combined with <code>--all</code> , uses a subread instead of a draft as representative consensus.
<code>--suppress-reports</code>	Suppresses the generation of default reports and metric files.

Interpreting QUAL values

The QUAL value of a read is a measure of the posterior likelihood of an error at a particular position. **Increasing** QUAL values are associated with a **decreasing** probability of error. For indels and homopolymers, there is ambiguity as to which QUAL value is associated with the error probability. Shown below are different types of alignment errors, with an * indicating which sequence BP should be associated with the alignment error.

Mismatch

```

      *
ccs:  ACGTATA
ref:  ACATATA

```

Deletion

```

      *
ccs:  AC-TATA
ref:  ACATATA

```

Insertion

```

      *
ccs:  ACGTATA
ref:  AC-TATA

```

Homopolymer insertion or deletion

Indels should always be left-aligned, and the error probability is only given for the first base in a homopolymer.

```

      *                      *
ccs:  ACGGGGTATA  ccs:  AC-GGGTATA
ref:  AC-GGGTATA  ref:  ACGGGGTATA

```

CCS Analysis Yield report

The CCS Analysis Yield report specifies the number of ZMWs that successfully produced consensus sequences, as well as a count of how many ZMWs did **not** produce a consensus sequence for various reasons. The entries in this report, as well as parameters used to increase or decrease the number of ZMWs that pass various filters, are shown in the table below.

The first part is a summary of inputs and outputs:

ZMW results	Parameters affecting results	Description
ZMWs input	None	The number of input ZMWs.
ZMWs pass filters	All custom processing settings	The number of CCS reads successfully produced on the first attempt, using the fast windowed approach.
ZMWs fail filters	All custom processing settings	The number of ZMWs reads that failed to produce a CCS read.
ZMWs shortcut filters	-all	The number of ZMWs having fewer than 2 full-length subreads.
ZMWs with tandem repeats	--min-tandem-repeat-length	The number of ZMWs with repeats larger than the specified threshold.

The second part explains in detail the exclusive ZMW count for those ZMWs that were filtered:

ZMW results	Parameters affecting results	Description
No usable subreads	None	The ZMW had no usable subreads. Either there were no subreads, or all subreads had lengths outside the range <50% or >200% of the median subread length.
Below SNR threshold	--min-snr	The ZMW had at least one channel's SNR below the minimum threshold.
Lacking full passes	--min-passes	There were not enough subreads that had an adapter at the start and end of the subread (a "full pass").
Heteroduplexes	None	The SMRTbell contains a heteroduplex. In this case, it is not clear what the consensus should be and so the ZMW is dropped.
Min coverage violation	None	The ZMW is damaged on one strand and cannot be polished reliably.
Draft generation error	None	Subreads do not match the generated draft sequence, even after multiple tries.
Draft above --max-length	--max-length	The draft sequence was above the maximum length threshold.
Draft below --min-length	--min-length	The draft sequence was below the minimum length threshold.
Lacking usable subreads	None	Too many subreads were dropped while polishing.
CCS analysis did not converge	None	The consensus sequence did not converge after the maximum number of allowed rounds of polishing.

ZMW results	Parameters affecting results	Description
CCS read below minimum predicted accuracy	<code>--min-rq</code>	Each CCS read has a predicted level of accuracy associated with it. Reads that are below the minimum specified threshold are removed.
Unknown error during processing	None	These should not occur.

How do I read the `ccs_report.txt` file?

By default, each CCS analysis generates a `ccs_report.txt` file. This file summarizes how many ZMWs generated HiFi reads and how many ZMWs failed CCS reads generation because of the listed causes. For those failing, each ZMW contributes to exactly one reason of failure; percentages are with respect to number of failed ZMWs.

Does CCS analysis dislike low-complexity regions?

Low-complexity comes in many shapes and forms. A particular challenge for CCS analysis are highly-enriched tandem repeats, such as hundreds of copies of AGGGGT. Prior to `ccs` v5.0, inserts with many copies of a small repeat likely did not generate a consensus sequence. Since `ccs` v5.0, every ZMW is tested if it contains a tandem repeat of length

`--min-tandem-repeat-length 1000`. For this, we use symmetric DUST, specifically [this](#) `sdust` implementation, but slightly modified. If a ZMW is flagged as a tandem repeat, internally `--disable-heuristics` is activated for only this ZMW, and various filters that are known to exclude low-complexity sequences are disabled. This recovers most of the low-complexity consensus sequences, without impacting run time performance.

Can I produce one consensus sequence for each strand of a molecule?

Yes, use `--by-strand`. Make sure that you have sufficient coverage, as `--min-passes` are per-strand in this case. For each strand, CCS analysis generates one consensus read that has to pass all filters. Read name suffix indicates strand. **Example:**

```
m64011_190714_120746/14/ccs/rev
m64011_190714_120746/35/ccs/fwd
```

How does `--by-strand` work? For each ZMW:

- Determine orientation of reads with respect to the one closest to the median length.
- Sort reads into two buckets, forward and reverse strands.
- Treat each strand as an individual entity as we do with ZMWs:
 - Apply all filters per strand individually.
 - Create a draft for each strand.

- Polish each strand.
- Write out each polished strand consensus.

BAM tags generated

Tag	Type	Description
ec	f	Effective coverage
fi	B,C	Forward IPD (Codec V1)
fn	i	Forward number of complete passes (zero or more)
fp	B,C	Forward PulseWidth (Codec V1)
np	i	Number of full-length subreads
ri	B,C	Reverse IPD (Codec V1)
rn	i	Reverse number of complete passes (zero or more)
rp	B,C	Reverse PulseWidth (Codec V1)
rq	f	Predicted average read accuracy
sn	B,F	Signal-to-noise ratios for each nucleotide
zm	i	ZMW hole number
RG	z	Read group

dataset The `dataset` tool creates, opens, manipulates and writes Data Set XML files. The commands allow you to perform operations on the various types of data held by a Data Set XML: Merge, split, write, and so on.

Usage

```
dataset [-h] [--version] [--log-file LOG_FILE]
        [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug | --quiet | -v]
        [--strict] [--skipCounts]
```

{create,filter,merge,split,validate,summarize,consolidate,loadstats,newuuid,loadmetadata,copyto,absolutize,relativize}

Options	Description
-h, --help	Displays help information and exits.
<Command> -h	Displays help for a specific command.
-v, --version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. (Default = None, writes to stdout.)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL]. (Default = INFO)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
-v	Sets the verbosity level. (Default = NONE)
--strict	Turns on strict tests and display all errors. (Default = False)
--skipCounts	Skips updating NumRecords and TotalLength counts. (Default = False)

`create` command: Create an XML file from a `fofn` (file-of-file names) or BAM file. Possible types: SubreadSet, AlignmentSet, ReferenceSet, HdfSubreadSet, BarcodeSet, ConsensusAlignmentSet, ConsensusReadSet, ContigSet.

```
dataset create [-h] [--type DSTYPE] [--name DSNAME] [--generateIndices]
               [--metadata METADATA] [--novalidate] [--relative]
               outfile infile [infile ...]
```

Example

The following example shows how to use the `dataset create` command to create a barcode file:

```
dataset create --generateIndices --name my_barcodes --type BarcodeSet
my_barcodes.barcodeset.xml my_barcodes.fasta
```

Required	Description
outfile	The name of the XML file to create.
infile	The <code>fofn</code> (file-of-file-names) or BAM file(s) to convert into an XML file.

Options	Description
--type DSTYPE	Specifies the type of XML file to create. (Default = NONE)
--name DSNAME	The name of the new Data Set XML file.
--generateIndices	Generates index files (.pbi and .bai for BAM, .fai for FASTA). Requires samtools/pysam and pbindex. (Default = FALSE)
--metadata METADATA	A metadata.xml file (or Data Set XML) to supply metadata. (Default = NONE)
--novalidate	Specifies not to validate the resulting XML. Leaves the paths as they are.
--relative	Makes the included paths relative instead of absolute. This is not compatible with --novalidate.

`filter` command: Filter an XML file using filters and threshold values.

- **Suggested filters:** `alignedlength`, `as`, `astart`, `bc`, `bcf`, `bcq`, `bcr`, `bq`, `cx`, `length`, `mapqv`, `movie`, `n_subreads`, `pos`, `qend`, `qid`, `qname`, `qstart`, `readstart`, `rname`, `rq`, `tend`, `tstart`, `zm`
- **More resource-intensive filter:** `[qs]`

Note: Multiple filters with different names are ANDed together. Multiple filters with the **same** name are ORed together, duplicating existing requirements. The filter string should be enclosed in **single quotes**.

```
dataset filter [-h] infile outfile filters [filters ...]
```

Required	Description
infile	The name of the XML file to filter.
outfile	The name of the output filtered XML file.
filters	The values to filter on. (Example: <code>rq>0.85</code>)

Examples

Filter on read quality >0.99 (Q20):

```
% dataset filter in.consensusreadset.xml hifi.consensusreadset.xml 'rq >= 0.99'
```

Filter on read quality and length:

```
% dataset filter in.consensusreadset.xml filtered.consensusreadset.xml 'rq >= 0.99 AND length >= 10000'
```

Filter for very long and very short reads:

```
% dataset filter in.consensusreadset.xml filtered.consensusreadset.xml 'length >= 40000; length <= 400'
```

Filter for specific high-quality barcodes:

```
% dataset filter mixed.consensusreadset.xml samples1-3.consensusreadset.xml 'bc = [0,1,2] AND bq >= 26'
```

merge command: Combine XML files.

```
dataset merge [-h] outfile infiles [infiles ...]
```

Required	Description
infiles	The names of the XML files to merge.
outfile	The name of the output XML file.

split command: Split a Data Set XML file.

```
dataset split [-h] [--contigs] [--barcodes] [--zmws] [--byRefLength]
               [--noCounts] [--chunks CHUNKS] [--maxChunks MAXCHUNKS]
               [--targetSize TARGETSIZE] [--breakContigs]
               [--subdatasets] [--outdir
               infile [outfiles...]
```

Required	Description
infile	The name of the XML file to split.

Options	Description
outfiles	The names of the resulting XML files.
--contigs	Splits the XML file based on contigs. (Default = FALSE)
--barcodes	Splits the XML file based on barcodes. (Default = FALSE)
--zmws	Splits the XML file based on ZMWs. (Default = FALSE)
--byRefLength	Splits contigs by contig length. (Default = TRUE)
--noCounts	Updates the Data Set counts after the split. (Default = FALSE)
--chunks x	Splits contigs into x total windows. (Default = 0)
--maxChunks x	Splits the contig list into at most x groups. (Default = 0)
--targetSize x	Specifies the minimum number of records per chunk. (Default = 5000)
--breakContigs	Breaks contigs to get closer to maxCounts. (Default = False)
--subdatasets	Splits the XML file based on sub-datasets. (Default = False)
--outdir OUTDIR	Specifies an output directory for the resulting XML files. (Default = <in-place>, not the current working directory.)

validate command: Validate XML and ResourceId files. (This is an internal testing functionality that may be useful.)

Note: This command requires that `pyxb` (**not** distributed with SMRT Link) be installed. If **not** installed, `validate` simply checks that the files pointed to in `ResourceIds` exist.


```
dataset validate [-h] [--skipFiles] infile
```

Required	Description
infile	The name of the XML file to validate.

Options	Description
--skipFiles	Skips validating external resources. (Default = False)

summarize command: Summarize a Data Set XML file.

```
dataset summarize [-h] infile
```

Required	Description
infile	The name of the XML file to summarize.

consolidate command: Consolidate XML files.

```
dataset consolidate [-h] [--numFiles NUMFILES] [--noTmp]
infile datafile xmlfile
```

Required	Description
infile	The name of the XML file to consolidate.
datafile	The name of the resulting data file.
xmlfile	The name of the resulting XML file.

Options	Description
--numFiles x	Specifies the number of data files to produce. (Default = 1)
--noTmp	Do not copy to a temporary location to ensure local disk use. (Default = False)

loadstats command: Load an sts.xml file containing pipeline statistics into a Data Set XML file.

```
dataset loadstats [-h] [--outfile OUTFILE] infile statsfile
```

Required	Description
infile	The name of the Data Set XML file to modify.
statsfile	The name of the sts.xml file to load.

Options	Description
--outfile OUTFILE	The name of the XML file to output. (Default = None)

`newuuid` command: Refresh a Data Set's Unique ID.

```
dataset newuuid [-h] [--random] infile
```

Required	Description
<code>infile</code>	The name of the XML file to refresh.

Options	Description
<code>--random</code>	Generates a random UUID, instead of a hash. (Default = <code>False</code>)

`loadmetadata` command: Load a `.metadata.xml` file into a Data Set XML file.

```
dataset loadmetadata [-h] [--outfile OUTFILE] infile metadata
```

Required	Description
<code>infile</code>	The name of the Data Set XML file to modify.
<code>metadata</code>	The <code>.metadata.xml</code> file to load, or Data Set to borrow from.

Options	Description
<code>--outfile OUTFILE</code>	Specifies the XML file to output. (Default = <code>None</code>)

`copyto` command: Copy a Data Set and resources to a new location.

```
dataset copyto [-h] [--relative] infile outdir
```

Required	Description
<code>infile</code>	The name of the XML file to copy.
<code>outdir</code>	The directory to copy to.

Options	Description
<code>--relative</code>	Makes the included paths relative instead of absolute. (Default = <code>False</code>)

`absolutize` command: Make the paths in an XML file absolute.

```
dataset absolutize [-h] [--outdir OUTDIR] infile
```

Required	Description
<code>infile</code>	The name of the XML file whose paths should be absolute.

Options	Description
<code>--outdir OUTDIR</code>	Specifies an optional output directory. (Default = <code>None</code>)

`relativize` command: Make the paths in an XML file relative.

`dataset relativize [-h] infile`

Required	Description
<code>infile</code>	The name of the XML file whose paths should be relative.

Examples - Filter reads

To filter one or more BAM file's worth of subreads, aligned or otherwise, and then place them into a single BAM file:

```
# usage: dataset filter <in_fn.xml> <out_fn.xml> <filters>
dataset filter in_fn.subreadset.xml filtered_fn.subreadset.xml 'rq>0.85'

# usage: dataset consolidate <in_fn.xml> <out_data_fn.bam> <out_fn.xml>
dataset consolidate filtered_fn.subreadset.xml consolidate.subreads.bam
out_fn.subreadset.xml
```

The filtered Data Set and the consolidated Data Set should be read-for-read equivalent when used with SMRT® Analysis software.

Example - Resequencing pipeline

- Align two movie's worth of subreads in two SubreadSets to a reference.
- Merge the subreads together.
- Split the subreads into Data Set chunks by contig.
- Process using `gcpp` on a chunkwise basis (in parallel).

1. Align each movie to the reference, producing a Data Set with one BAM file for each execution:

```
pbalign movie1.subreadset.xml referenceset.xml movie1.alignmentset.xml
pbalign movie2.subreadset.xml referenceset.xml movie2.alignmentset.xml
```

2. Merge the files into a FOFN-like Data Set; BAMs are not touched:

```
# dataset merge <out_fn> <in_fn> [<in_fn> <in_fn> ...]
dataset merge merged.alignmentset.xml movie1.alignmentset.xml movie2.alignmentset.xml
```

3. Split the Data Set into chunks by contig name; BAMs are not touched:
 - Note that supplying output files splits the Data Set into that many output files (up to the number of contigs), with multiple contigs per file.
 - **Not** supplying output files splits the Data Set into **one** output file per contig, named automatically.
 - Specifying a number of chunks instead will produce that many files, with contig or even subcontig (reference window) splitting.

```
dataset split --contigs --chunks 8 merged.alignmentset.xml
```

4. Process the chunks:

```
gcpp --reference referenceset.xml --output  
chunk1consensus.fasta,chunk1consensus.fastq,chunk1consensus.vcf,chunk1consensus.gff  
chunk1contigs.alignmentset.xml
```

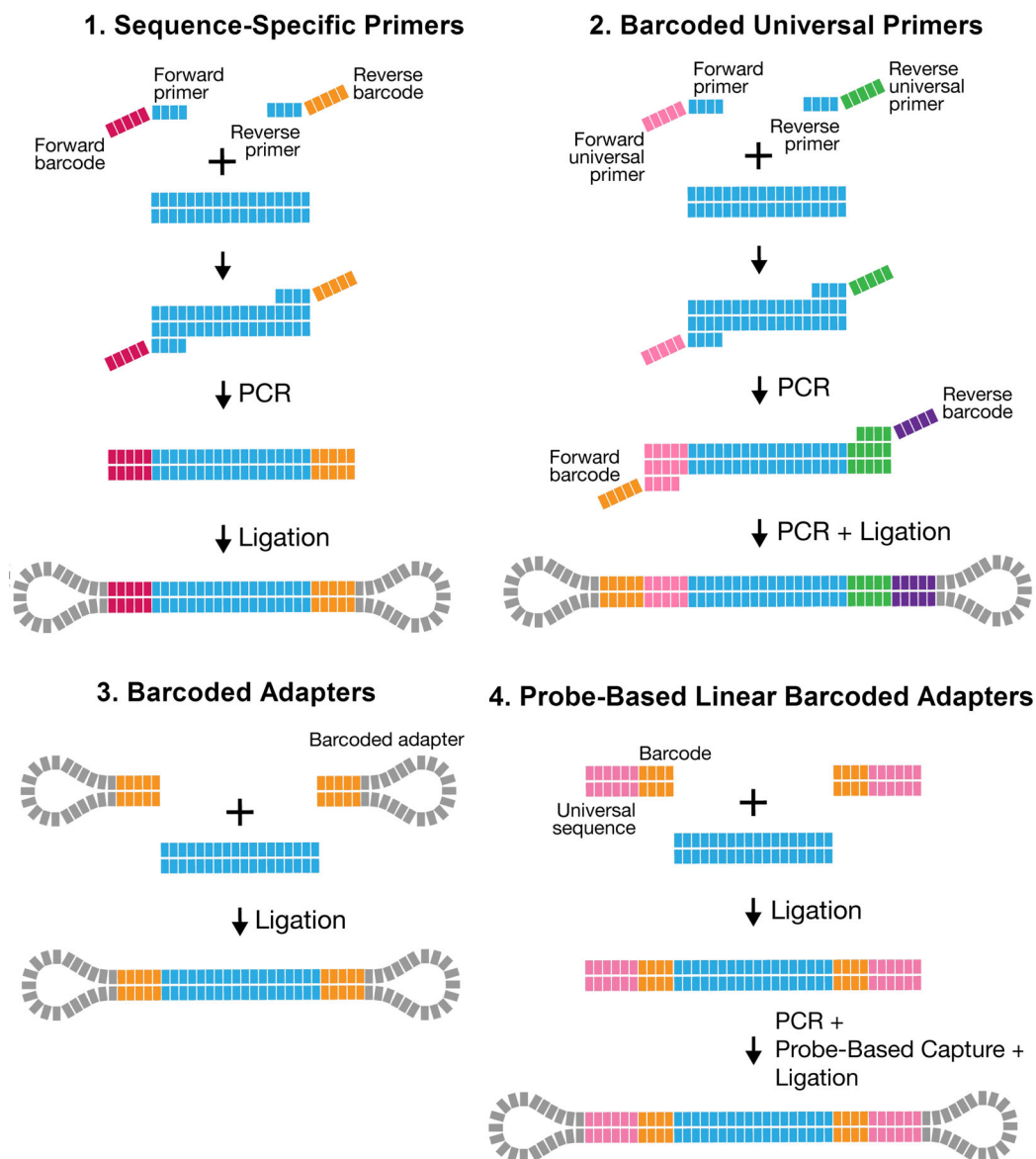
The chunking works by duplicating the original merged Data Set (no BAM duplication) and adding filters to each duplicate such that only reads belonging to the appropriate contigs are emitted. The contigs are distributed among the output files in such a way that the total number of records per chunk is about even.

Demultiplex Barcodes

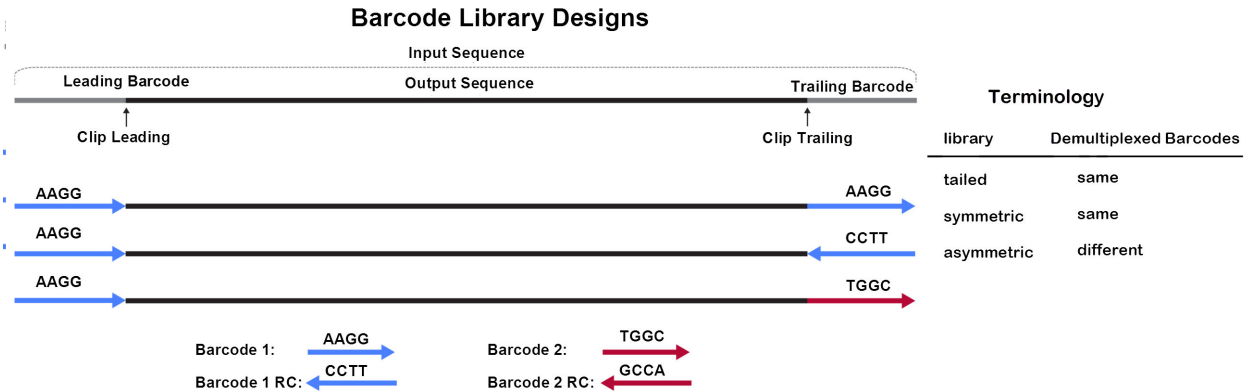
The **Demultiplex Barcodes** application identifies barcode sequences in PacBio single-molecule sequencing data.

Demultiplex Barcodes can demultiplex samples that have a unique per-sample barcode pair and were pooled and sequenced on the same SMRT® Cell. There are four different methods for barcoding samples with PacBio technology:

1. Sequence-specific primers
2. Barcoded universal primers
3. Barcoded adapters
4. Linear Barcoded Adapters for Probe-based Captures



In addition, there are three different barcode library designs. **Demultiplex Barcodes** supports raw subread and CCS reads demultiplexing.



In the overview above, the input sequence is flanked by adapters on both sides. The bases adjacent to an adapter are **barcode regions**. A read can have up to two barcode regions, leading and trailing. Either or both adapters can be missing and consequently the leading and/or trailing region is not being identified.

For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For barcode identification, one read with a single barcode region is sufficient.

For the **asymmetric** design, **different** barcodes are attached to the sides of the insert sequence of interest. To identify the different barcodes, a read with leading and trailing barcode regions is required.

Output barcode pairs are generated from the identified barcodes. The barcode names are combined using "--", for example `bc1002--bc1054`. The sort order is defined by the barcode indices, starting with the lowest.

Workflow

By default, **Demultiplex Barcodes** processes input reads grouped by ZMW, **except** if the `--per-read` option is used. All barcode regions along the read are processed individually. The final per-ZMW result is a summary over all barcode regions. Each ZMW is assigned to a pair of selected barcodes from the provided set of candidate barcodes. Subreads from the same ZMW will have the same barcode and barcode quality. For a particular target barcode region, every barcode sequence gets aligned as given and as reverse-complement, and higher scoring orientation is chosen. This results in a list of scores over all candidate barcodes.

- If only **same** barcode pairs are of interest (symmetric/tailed), use the `--same` option to filter out **different** barcode pairs.

- If only **different** barcode pairs are of interest (asymmetric), use the `--different` option to require at least two barcodes to be read, and remove pairs with the **same** barcode.

Parameter presets

Recommended parameter combinations are available using `--preset` for HiFi input:

- **HIFI-SYMMETRIC**
`--ccs --min-score 80 --min-end-score 50 --min-ref-span 0.75 --same`
- **HIFI-ASYMMETRIC**
`--ccs --min-score 80 --min-end-score 50 --min-ref-span 0.75 --different --min-scoring-regions 2`
- **NONE (Default)**

Half adapters

For an adapter call with only one barcode region, the high-quality region finder cuts right through the adapter. The preceding or succeeding subread was too short and was removed, or the sequencing reaction started/stopped there. This is called a **half adapter**. Thus, there are also 1.5, 2.5, N+0.5 adapter calls.

ZMWs with half or only one adapter can be used to identify the same barcode pairs; positive-predictive value might be reduced compared to high adapter calls. For asymmetric designs with different barcodes in a pair, at least a single full-pass read is required; this can be two adapters, two half adapters, or a combination.

Usage

- Any existing output files are **overwritten** after execution.
- Always use `--peek-guess` to remove spurious barcode hits.

Analysis of subread data

```
lima movie.subreads.bam barcodes.fasta prefix.bam
lima movie.subreadset.xml barcodes.barcodeset.xml prefix.subreadset.xml
```

Analysis of CCS reads

```
lima --css movie.ccs.bam barcodes.fasta prefix.bam
lima --ccs movie.consensusreadset.xml barcodes.barcodeset.xml
prefix.consensusreadset.xml
```

If you do not need to import the demultiplexed data into SMRT Link, use the `--no-pbi` option to minimize memory consumption and run time.

Symmetric or tailed options

```
Raw: --same
CCS read: --same --ccs
```

Asymmetric options

Raw: --different

CCS reads: --different --ccs

Example execution

```
lima m54317_180718_075644.subreadset.xml \
Sequel_RSII_384_barcode_v1.barcode.v1 \
m54317_180718_075644.demux.subreadset.xml \
--different --peek-guess
```

Options	Description
--same	Retains only reads with the same barcodes on both ends of the insert sequence, such as symmetric and tailed designs.
--different	Retains only reads with different barcodes on both ends of the insert sequence, asymmetric designs. Enforces <code>--min-passes ≥ 1</code> .
--min-length n	Omits reads with lengths below n base pairs after demultiplexing. ZMWs with no reads passing are omitted. (Default = 50)
--max-input-length n	Omits reads with lengths above n base pairs for scoring in the demultiplexing step. (Default = 0, deactivated)
--min-score n	Omits ZMWs with average barcode scores below n. A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator how well the chosen barcode pair matches. It is normalized to a range between 0 (no hit) and 100 (a perfect match). (Default = 0, PacBio recommends setting it to 26.)
--min-end-score n	Specifies the minimum end barcode score threshold applied to the individual leading and trailing ends. (Default = 0)
--min-passes n	Omits ZMWs with less than n full passes, a read with a leading and trailing adapter. (Default = 0, no full-pass needed) Example: 0 pass : insert - adapter - insert 1 pass : insert - adapter - INSERT - adapter - insert 2 passes: insert - adapter - INSERT - adapter - INSERT - adapter - insert
--score-full-pass	Uses only reads flanked by adapters on both sides (full-pass reads) for barcode identification.
--min-ref-span	Specifies the minimum reference span relative to the barcode length. (Default = 0.5)
--per-read	Scores and tags per subread, instead of per ZMW.
--ccs	Sets defaults to -A 1 -B 4 -D 3 -I 3 -X 1.
--peek n	Looks at the first n ZMWs of the input and return the mean. This lets you test multiple test barcode.fasta files and see which set of barcodes was used.
--guess n	This performs demultiplexing twice. In the first iteration, all barcodes are tested per ZMW. Afterwards, the barcode occurrences are counted and their mean is tested against the threshold n; only those barcode pairs that pass this threshold are used in the second iteration to produce the final demultiplexed output. A prefix.lima.guess file shows the decision process; --same is being respected.
--guess-min-count	Specifies the minimum ZMW count to whitelist a barcode. This filter is ANDed with the minimum barcode score specified by --guess. (Default = 0)

Options	Description
<code>--peek-guess</code>	Sets the following options: <code>--peek 50000 --guess 45 --guess-min-count 10</code> . Demultiplex Barcodes will run twice on the input data. For the first 50,000 ZMWs, it will guess the barcodes and store the mask of identified barcodes. In the second run, the barcode mask is used to demultiplex all ZMWs. If combined with <code>--ccs</code> then the barcode score threshold is increased by <code>--guess 75</code> .
<code>--single-side</code>	Identifies barcodes in molecules that only have barcodes adjacent to one adapter.
<code>--window-size-mult</code> <code>--window-size-bp</code>	The candidate region size multiplier: <code>barcode_length * multiplier</code> . (Default = 3) Optionally, you can specify the region size in base pairs using <code>--window-size-bp</code> . If set, <code>--window-size-mult</code> is ignored.
<code>--num-threads n</code>	Spawns <code>n</code> threads; 0 means use all available cores. This option also controls the number of threads used for BAM and PBI compression. (Default = 0)
<code>--chunk-size n</code>	Specifies that each thread consumes <code>n</code> ZMWs per chunk for processing. (Default = 10).
<code>--no-bam</code>	Does not produce BAM output. Useful if only reports are of interest, as run time is shorter.
<code>--no-pbi</code>	Does not produce a <code>.bam.pbi</code> index file. The on-the-fly <code>.bam.pbi</code> file generation buffers the output data. If you do not need a <code>.bam.pbi</code> index file for SMRT Link import, use this option to decrease memory usage to a minimum and shorten the run time.
<code>--no-reports</code>	Does not produce any reports. Useful if only demultiplexed BAM files are needed.
<code>--dump-clips</code>	Outputs all clipped barcode regions generated to the <code><prefix>.lima.clips</code> file.
<code>--dump-removed</code>	Outputs all records that did not pass the specified thresholds, or are without barcodes, to the <code><prefix>.lima.removed.bam</code> file.
<code>--split-bam</code> <code>--split-bam-named</code>	Specifies that each barcode has its own BAM file called <code>prefix.idxBest-idxCombined.bam</code> , such as <code>prefix.0-0.bam</code> . Optionally, <code>--split-bam-named</code> names the files by their barcode names instead of their barcode indices.
<code>--isoseq</code>	Removes primers as part of the Iso-Seq® pipeline. See “Demultiplexing Iso-Seq® data” on page 33 for details.
<code>--bad-adapter-ratio n</code>	Specifies the maximum ratio of bad adapters. (Default = 0).

Input files

Input data in PacBio-enhanced BAM format is either:

- Sequence data - Unaligned subreads, directly from Sequel II systems and Sequel IIe systems.
- Unaligned CCS reads, generated by CCS analysis.

Barcodes are provided as a FASTA file or BarcodeSet file:

- One entry per barcode sequence.
- **No** duplicate sequences.

- All bases must be in **upper-case**.
- Orientation-agnostic (forward or reverse-complement, but **not** reversed.)

Example

```
>bc1000
CTCTACTTACTTACTG
>bc1001
GTCGTATCATCATGTA
>bc1002
AATATACCTATCATTA
```

Note: Name barcodes using an alphabetic character prefix to avoid later barcode name/index confusion.

Output files

Demultiplex Barcodes generates multiple output files by default, all starting with the same prefix as the output file, using the suffixes `.bam`, `.subreadset.xml`, and `.consensusreadset.xml`. The report prefix is `lima`. **Example:**

```
lima m54007_170702_064558.subreads.bam barcode.fasta /my/path/
m54007_170702_064558_demux.subreadset.xml
```

For all output files, the prefix is

`/my/path/m54007_170702_064558_demux.`

- `<prefix>.bam`: Contains clipped records, annotated with barcode tags, that passed filters and respect the `--same` option.
- `<prefix>.lima.report`: A tab-separated file describing each ZMW, unfiltered. This is useful information for investigating the demultiplexing process and the underlying data. A single row contains **all** reads from a single ZMW. For `--per-read`, each row contains one subread, and ZMWs might span multiple rows.
- `<prefix>.lima.summary`: Lists how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.

(1)

```
ZMWs input (A): 213120
ZMWs above all thresholds (B): 176356 (83%)
ZMWs below any threshold (C): 36764 (17%)
```

(2)

```
ZMW Marginals for (C):
Below min length : 26 (0%)
Below min score : 0 (0%)
Below min end score : 5138 (13%)
Below min passes : 0 (0%)
Below min score lead : 11656 (32%)
Below min ref span : 3124 (8%)
Without adapter : 25094 (68%)
With bad adapter : 10349 (28%) <- Only with --bad-adapter-ratio
```

```

Undesired hybrids           : xxx (xx%) <- Only with --peek-guess
Undesired same barcode pairs : xxx (xx%) <- Only with --different
Undesired diff barcode pairs : xxx (xx%) <- Only with --same
Undesired 5p--5p pairs      : xxx (xx%) <- Only with --isoseq
Undesired 3p--3p pairs      : xxx (xx%) <- Only with --isoseq
Undesired single side       : xxx (xx%) <- Only with --isoseq
Undesired no hit            : xxx (xx%) <- Only with --isoseq

```

(3)

```

ZMWs for (B):
With same barcode           : 162244 (92%)
With different barcodes     : 14112 (8%)
Coefficient of correlation   : 32.79%

```

(4)

```

ZMWs for (A):
Allow diff barcode pair     : 157264 (74%)
Allow same barcode pair     : 188026 (88%)
Bad adapter yield loss      : 10112 (5%) <- Only with --bad-adapter-ratio
Bad adapter impurity        : 10348 (5%) <- Only without --bad-adapter-ratio

```

(5)

```

Reads for (B):
Above length                : 1278461 (100%)
Below length                : 2787 (0%)

```

Explanation of each block:

1. Number of ZMWs that went into `lima`, how many ZMWs were passed to the output file, and how many did not qualify.
2. For those ZMWs that did not qualify: The marginal counts of each filter. (Filter are described in the Options table.)
When running with `--peek-guess` or similar manual option combination and different barcode pairs are found during peek, the full SMRT Cell may contain low-abundant different barcode pairs that were identified during peek individually, but not as a pair. Those unwanted barcode pairs are called hybrids.
3. For those ZMWs that passed: How many were flagged as having the same or different barcode pair, as well as the coefficient of variation for the barcode ZMW yield distribution in percent.
4. For all input ZMWs: How many allow calling the same or different barcode pair. This is a simplified version of how many ZMW have at least one full pass to allow a different barcode pair call and how many ZMWs have at least half an adapter, allowing the same barcode pair call.
5. For those ZMWs that qualified: The number of reads that are above and below the specified `--min-length` threshold.
 - `<prefix>.lima.counts: A .tsv` file listing the counts of each observed barcode pair. Only passing ZMWs are counted.

Example: `column -t prefix.lima.count`

IdxFirst	IdxCombined	IdxFirstNamed	IdxCombinedNamed	Counts	MeanScore
0	0	bc1001	bc1001	1145	68
1	1	bc1002	bc1002	974	69
2	2	bc1003	bc1003	1087	68

- `<prefix>.lima.clips`: Contains clipped barcode regions generated using the `--dump-clips` option. **Example:**

```
head -n 6 prefix.lima.clips
>m54007_170702_064558/4850602/6488_6512 bq:34 bc:11
CATGTCCCTCAGTTAAGTTACAA
>m54007_170702_064558/4850602/6582_6605 bq:37 bc:11
TTTGTACTAAGTATACCAATAG
>m54007_170702_064558/4916040/4801_4816 bq:93 bc:10
```
- `<prefix>.lima.removed.bam`: Contains records that did **not** pass the specified thresholds, or are without barcodes, using the option `--dump-removed.lima` does **not** generate a `.pbi`, nor Data Set for this file. This option **cannot** be used with any splitting option.
- `<prefix>.lima.guess`: A `.tsv` file that describes the barcode subsetting process activated using the `--peek` and `--guess` options.

IdxFirst	IdxCombined	IdxFirstNamed	IdxCombinedNamed	NumZMWs	MeanScore	Picked
0	0	bc1001t	bc1001t	1008	50	1
1	1	bc1002t	bc1002t	1005	60	1
2	2	bc1003t	bc1003t	5	24	0
3	3	bc1004t	bc1004t	555	61	1

- One `DataSet`, `.subreadset.xml`, or `.consensusreadset.xml` file is generated per output BAM file.
- `.pbi`: One PBI file is generated per output BAM file.

What is a universal spacer sequence and how does it affect demultiplexing?

For library designs that include an identical sequence between adapter and barcode, such as probe-based linear barcoded adapters samples, Demultiplex Barcodes offers a special mode that is activated if it finds a shared prefix sequence among all provided barcode sequences.

Example

```
>custombc1
ACATGACTGTGACTATCTCACACATATCAGAGTGCG
>custombc2
ACATGACTGTGACTATCTCAACACACAGACTGTGAG
```

In this case, Demultiplex Barcodes detects the shared prefix

ACATGACTGTGACTATCTCA and removes it internally from all barcodes. Subsequently, it increases the window size by the length L of the prefix sequence.

- If `--window-size-bp N` is used, the actual window size is $L + N$.
- If `--window-size-mult M` is used, the actual window size is $(L + |bc|) * M$.

Because the alignment is semi-global, a leading reference gap can be added without any penalty to the barcode score.

What are bad adapters?

In the `subreads.bam` file, each subread has a context flag `cx`. The flag specifies, among other things, whether a subread has flanking adapters, before and/or after. Adapter-finding was improved and can also find molecularly-missing adapters, or those obscured by a local decrease in accuracy. This may lead to missing or obscured bases in the flanking barcode. Such adapters are labelled "bad", as they don't align with the adapter reference sequence(s). Regions flanking those bad adapters are problematic, because they can fully or partially miss the barcode bases, leading to wrong classification of the molecule. `lima` can handle those adapters by **ignoring** regions flanking bad adapters. For this, `lima` computes the ratio of number of bad adapters divided by number of all adapters.

By default, `--bad-adapter-ratio` is set to 0 and does **not** perform any filtering. In this mode, bad adapters are handled just like good adapters.

But the `*.lima.summary` file contains one row with the number of ZMWs that have at least 25% bad adapters, but otherwise pass all other filters. This metric can be used as a diagnostic to assess library preparation.

If `--bad-adapter-ratio` is set to non-zero positive $(0, 1)$, bad adapter flanking barcode regions are treated as missing. If a ZMW has a higher ratio of bad adapters than provided, the ZMW is filtered and consequently removed from the output. The `*.lima.summary` file contains two additional rows.

```
With bad adapter      : 10349 (28%)
Bad adapter yield loss : 10112 (5%)
```

The first row counts the number of ZMWs that have bad adapter ratios that are too high; the percentage is with respect to the number of all ZMW not passing. The second row counts the number of ZMWs that are removed solely due to bad adapter ratios that are too high; the percentage is with respect the number of all input ZMWs and consequently is the effective yield loss caused by bad adapters.

If a ZMW has ~50% bad adapters, one side of the molecule is molecularly-missing an adapter. For 100% bad adapter, **both** sides are missing adapters. A lower than ~40% percentage indicates decreased local accuracy during sequencing leading to adapter sequences not being found. If a high percentage of ZMWs is molecularly-missing adapters, you should improve library preparation.

Demultiplexing Iso-Seq® data

Demultiplex Barcodes is used to identify and remove Iso-Seq cDNA primers. If the Iso-Seq sample is barcoded, the barcodes should be included as part of the primer. Only by using the command-line can users use `lima` with the `--isoseq` option for demultiplexing Iso-Seq data.

The input Iso-Seq data format for demultiplexing is `.ccs.bam`. Users must first generate a CCS reads BAM file for an Iso-Seq Data Set before running `lima`. The recommended parameters for running CCS analysis for Iso-Seq are `min-pass=1,min accuracy=0.9`, and turning Polish to OFF.

1. Primer IDs must be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should not include the Ts and is written in reverse complement.
2. Below are four example primer sets. The first is unbarcoded, the second has barcodes (shown in lower case) adjacent to the 3' primer.

Example 1: The Iso-Seq cDNA Primer primer set, included with the SMRT Link installation.

Users following the standard Iso-Seq Express protocol **without** multiplexing, or running a Data Set that has **already** been demultiplexed (either using Run Design or the SMRT® Analysis application) should use this default option.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>IsoSeq_3p
GTACTCTGCGTTGATACCACTGCTT
```

Example 2: The Iso-Seq 12 Barcoded cDNA Primers set, included with the SMRT Link installation.

Users using barcoded cDNA primers listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document

Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell Prep Kit 3.0, should select this option.

```
>bc1001_5p
CACATATCAGAGTGC GGCAATGAAGTCGCAGGGTTGGGG
>bc1002_5p
ACACACAGACTGTGAGGCAATGAAGTCGCAGGGTTGGGG
...
```

(There are a total of 24 sequence records, representing 12 pairs of F/R barcoded cDNA primers.)

Example 3: An example of a custom cDNA primer set. 4 tissues were multiplexed using barcodes on the 3' end only.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>dT_BC1001_3p
AAGCAGTGGTATCAACGCAGAGTACCACATATCAGAGTGC
>dT_BC1002_3p
AAGCAGTGGTATCAACGCAGAGTACACACACAGACTGTGAG
>dT_BC1003_3p
AAGCAGTGGTATCAACGCAGAGTACACACATCTCGTGAGAG
>dT_BC1004_3p
AAGCAGTGGTATCAACGCAGAGTACCACGCACACACGCGCG
```

Example 4: Special Handling for the TeloPrime cDNA Kit

The Lexogen TeloPrime cDNA kit contains **As** in the 3' primer that **cannot** be differentiated from the polyA tail. For best results, remove the **As** from the 3' end as shown below:

```
>TeloPrimeModified_5p
TGGATTGATATGTAATACGACTCACTATAG
>TeloPrimeModified_3p
CGCCTGAGA
```

3. Use the `--isoseq` mode. Note that this **cannot** be combined with the `--guess` option.
4. The output will be only different pairs with a 5p and 3p combination:

```
demux.5p--tissue1_3p.bam
demux.5p--tissue2_3p.bam
```

The `--isoseq` parameter set is very conservative for removing any spurious and ambiguous calls, and guarantees that only proper asymmetric (barcoded) primer are used in downstream analyses. Good libraries reach >75% CCS reads passing the Demultiplex Barcodes filters.

BAM tags

In SMRT Link v11.1, `LB` and `SM` tags are set by the user in Run Design. The `SM` tag can also be set in Demultiplex Barcodes in SMRT Analysis.

Non-demultiplex case:

- LB: Well Sample Name.
- SM: Bio Sample Name.

Multiplexed case, BAM pre-demultiplexing:

- LB: Well Sample Name.
- SM: Tag removed.

Multiplexed case, BAMs post-demultiplexing:

- LB: Well Sample Name for all child barcode BAMs.
- SM: Each individual Bio Sample Name for the specific barcode.
- BC: Barcode sequence or hyphenated barcode sequences of the pair.
- DS: Appends barcode information used in demultiplexing: BarcodeFile, BarcodeHash, BarcodeCount, BarcodeMode, BarcodeQuality.
- Example read group header after demultiplexing:

```
@RG
ID:66d5a6af/3--3
PL:PACBIO
DS:READTYPE=SUBREAD;
  Ipd:CodecV1=ip;
  PulseWidth:CodecV1=pw;
  BINDINGKIT=101-500-400;
  SEQUENCINGKIT=101-427-800;
  BASECALLERVERSION=5.0.0;
  FRAMERATEHZ=100.000000;
  BarcodeFile=Sequel_16_barcode_v3.barcode.set.xml;
  BarcodeHash=f2b1fa0b43eb6ccbb30749883bb550e3;
  BarcodeCount=16;
  BarcodeMode=Symmetric;
  BarcodeQuality=Score
PU:m54010_200212_162236
SM:MySampleName
PM:SEQUEL
BC:ACAGTCGAGCGCTGCGT
```


export-datasets The `export-datasets` tool takes one or more PacBio Data Set XML files and packages all contents (including index files and supplemental Data Sets) into a single ZIP archive. Data Set resources, such as BAM files, are reorganized and renamed to flatten the directory structure, avoid redundant file writes, and convert all resource paths from absolute paths to relative paths. Where multiple Data Sets are provided, the contents of each is nested in a directory named after the `UniqueId` attribute in the XML.

The resulting archive is primarily intended to be directly imported into SMRT Link using the Data Management interface, but it may also be unpacked manually and used on the command line.

Usage

```
export-datasets [options] <dataset>...
```

Options	Description
<code>-o, --output</code>	Name of output ZIP file. (Default = <code>datasets_<timestamp>.zip</code>)
<code>--keep-parent-ref</code>	Keeps the reference to the parent Data Set when archiving a demultiplexed child Data Set.
<code>--no-scrap</code>	Excludes the <code>scrap.bam</code> file if present in the XML file.
<code>-h, --help</code>	Displays help information and exits.
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are <code>[ERROR, DEBUG, INFO, WARN]</code> . (Default = <code>WARN</code>)
<code>--logback</code>	Override all logger configuration using a specified <code>logback.xml</code> file.
<code>--log2stdout</code>	If <code>True</code> , log output is displayed to the console. (Default = <code>False</code>)
<code>--debug</code>	Alias for setting the log level to <code>DEBUG</code> . (Default = <code>False</code>)
<code>--quiet</code>	Alias for setting the log level to <code>ERROR</code> . (Default = <code>False</code>)
<code>--verbose</code>	Alias for setting the log level to <code>INFO</code> . (Default = <code>False</code>)

Input files

- One or more PacBio Dataset XML files.

Output file

- One output ZIP file.

Examples

```
export-datasets m64001_200704_012345.subreadset.xml
```

```
export-datasets sample1.consensusreadset.xml sample2.consensusreadset.xml
\sample3.consensusreadset.xml -o barcoded_ccs.zip
```

```
export-datasets /opt/smrtlink/jobs/0000/0000001/0000001234/outputs/
mapped.alignmentset.xml
```

export-job The `export-job` tool packages a SMRT Link Analysis job for export to another system, usually for reimportation into another SMRT Link instance. All internal paths in job output files are converted from absolute to relative paths, and many of the internal details of the Cromwell workflows are omitted. The export is **not** a complete record of the job, but rather a collection of job output files and metadata.

Note that `export-job` **will** include any external Data Sets referenced in output Data Sets inside the job, for example ReferenceSets associated with mapped Data Sets, or BarcodeSets associated with demultiplexed Data Sets. However, these Data Sets will **not** be imported along with the job. The exported job does **not** include the input reads used to run the job; these may be exported separately using the `export-datasets` tool.

Important: Only SMRT Link v10.0 or later generates the necessary metadata files for `export-job` to save a full record of job execution. Jobs created with older versions of SMRT Link will still be archived, but the metadata will be empty and/or incorrect.

Usage

```
export-job [options] <job_dir>
```

Options	Description
<job_dir>	Path to a SMRT Link job directory.
-o, --output	Name of output ZIP file. (Default = job_<timestamp>.zip)
-h, --help	Displays help information and exits.
--log-file	Writes the log to a file. (Default = stderr)
--log-level	Specifies the log level; values are [ERROR, DEBUG, INFO, WARN]. (Default = WARN)
--logback	Override all logger configuration using a specified logback.xml file.
--log2stdout	If True, log output is displayed to the console. (Default = False)
--debug	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to ERROR. (Default = False)
--verbose	Alias for setting the log level to INFO. (Default = False)

Input

- A path to a job directory.

Output file

- One output ZIP file.

Examples

```
export-job /path/to/smrtlink/jobs-root/0000/0000000/00000000860 -o job860.zip
```

To reimport on another system:

```
pbservice import-job job860.zip
```

gcpp gcpp is a variant-calling tool provided by the GCpp package which provides several variant-calling algorithms for PacBio sequencing data.

Usage

```
gcpp      -j8 --algorithm=arrow \
          -r lambdaNEB.fa        \
          -o variants.gff        \
          aligned_subreads.bam
```

This example requests variant-calling, using 8 worker processes and the Arrow algorithm, taking input from the file `aligned_subreads.bam`, using the FASTA file `lambdaNEB.fa` as the reference, and writing output to `variants.gff`.

A particularly useful option is `--referenceWindow/-w`; which allows the variant-calling to be performed exclusively on a **window** of the reference genome.

Input files

- A sorted file of reference-aligned reads in PacBio's standard BAM format.
- A FASTA file that follows the PacBio FASTA file convention. If specifying an input FASTA file, a FASTA index file (`.fai`) with the same name and path is **required**. If the `.fai` file is not supplied, gcpp exits and displays an error message.

Note: The `--algorithm=arrow` option requires that certain metrics be in place in the input BAM file. It requires per-read SNR metrics, and the per-base `PulseWidth` metric for Sequel data.

The selected algorithm will stop with an error message if any features that it requires are unavailable.

Output files

Output files are specified as comma-separated arguments to the `-o` flag. The file name extension provided to the `-o` flag is meaningful, as it determines the output file format. For example:

```
gcpp aligned_subreads.bam -r lambda.fa -o myVariants.gff,myConsensus.fasta
```

will read input from `aligned_subreads.bam`, using the reference `lambda.fa`, and send variant call output to the file `myVariants.gff`, and consensus output to `myConsensus.fasta`.

The file formats currently supported (using extensions) are:

- `.gff`: PacBio GFFv3 variants format; convertible to BED.
- `.vcf`: VCF 4.2 variants format (that is compatible with v4.3.)
- `.fasta`: FASTA file recording the consensus sequence calculated for each reference contig.

- `.fastq`: FASTQ file recording the consensus sequence calculated for each reference contig, as well as per-base confidence scores.

Options	Description
<code>-j</code>	Specifies the number of worker processes to use.
<code>--algorithm=</code>	Specifies the variant-calling algorithm to use; values are <code>plurality</code> , <code>arrow</code> and <code>poa</code> . (Default = <code>arrow</code>)
<code>-r</code>	Specifies the FASTA reference file to use.
<code>-o</code>	Specifies the output file format; values are <code>.gff</code> , <code>.vcf</code> , <code>.fasta</code> , and <code>.fastq</code> .
<code>--maskRadius</code>	When using the <code>arrow</code> algorithm, setting this option to a value <code>N</code> greater than 0 causes <code>gcpp</code> to pass over the data a second time after masking out regions of reads that have >70% errors in $2*N+1$ bases. This setting has little to no effect at low coverage, but for high-coverage datasets (>50X), setting this parameter to 3 may improve final consensus accuracy. In rare circumstances, such as misassembly or mapping to the wrong reference, enabling this parameter may cause worse performance.
<code>--minConfidence MINCONFIDENCE</code> <code>-q MINCONFIDENCE</code>	Specifies the minimum confidence for a variant call to be output to variants.{gff,vcf} (Default = 40)
<code>--minCoverage MINCOVERAGE</code> <code>-x MINCOVERAGE</code>	Specifies the minimum site coverage for variant calls and consensus to be calculated for a site. (Default = 5)

Available algorithms

At this time there are three algorithms available for variant calling: `plurality`, `poa` and `arrow`.

- `plurality` is a simple and very fast procedure that merely tallies the most frequent read base or bases found in alignment with each reference base, and reports deviations from the reference as potential variants. This approach is prone to insertion and deletion errors.
- `poa` uses the partial order alignment algorithm to determine the consensus sequence. It is a heuristic algorithm that approximates a multiple sequence alignment by progressively aligning sequences to an existing set of alignments.
- `arrow` uses the per-read SNR metric and the per-pulse `pulsewidth` metric as part of its likelihood model.

Confidence values

The `arrow` and `plurality` algorithms make a confidence metric available for every position of the consensus sequence. The confidence should be interpreted as a phred-transformed posterior probability that the consensus call is incorrect; such as:

$$QV = -10\log_{10}(p_{err})$$

`gcpp` clips reported QV values at 93; larger values **cannot** be encoded in a standard FASTQ file.

Chemistry specificity

The `--algorithm=arrow` parameter is trained per-chemistry. `arrow` identifies the sequencing chemistry used for each run by looking at metadata contained in the input BAM data file. This behavior can be overridden by a command-line option.

When multiple chemistries are represented in the reads in the input file, the Arrow will model reads appropriately using the parameter set for its chemistry, thus yielding optimal results.

Genome Assembly

The Genome Assembly application generates *de novo* assemblies using HiFi reads. The application is fast, produces contiguous assemblies, and is suitable for genomes of any size.

The Genome Assembly application is powered by the IPA HiFi genome assembler and includes the following features:

- Separates haplotypes during assembly using a novel phasing stage (Nighthawk).
- Polishes the contigs with phased reads using `Racon`.
- Improves haplotype separation using the `purge_dups` tool.

Workflow of the Genome Assembly application

Analysis steps are highly optimized to produce assemblies of large genomes efficiently.



The workflow consists of seven stages:

1. Sequence database construction.
2. Fast overlap computation using the `Pancake` tool.
3. A dedicated phasing stage using the `Nighthawk` tool.
4. Filtering chimeras and residual repeats.
5. Layout based on the string graph.
6. Polishing using the `Racon` tool.
7. Purging haplotype duplicates from the primary assembly using the third-party tool `purge_dups`.

The workflow accepts HiFi XML Data Sets as input.

IPA HiFi genome assembler

- Scales well on a cluster.
- The workflow has an embedded downsampling feature:
 - If the genome size and the desired coverage are specified, the initial stage (sequence database construction) downsamples the input Data Set to the desired coverage.
 - Otherwise, the full coverage is used.

Usage

The Genome Assembly application is run using the `pbcrumwell run` command, with the `pb_assembly_hifi` parameter to specify the application. See [“pbcrumwell” on page 80](#) for details.

To view information on the available Genome Assembly options, enter:

```
pbcrumwell show-workflow-details pb_assembly_hifi
```

The **minimum** command needed to run the workflow requires the input and the number of threads. The following example uses 16 threads:

```
pbcrumwell run pb_assembly_hifi -e <input.xml> --nproc 16
```

The following example performs an assembly using an input XML Data Set, and uses all default settings, including 1 CPU:

```
pbcrumwell run pb_assembly_hifi -e <input.consensusreadset.xml>
```

Note: The default options for this workflow are equivalent to the following command:

```
pbcrumwell run pb_assembly_hifi \  
-e <input.consensusreadset.xml> \  
--task-option reads=None \  
--task-option ipa2_genome_size=0 \  
--task-option ipa2_downsampled_coverage=0 \  
--task-option ipa2_advanced_options="" \  
--task-option ipa2_run_polishing=True \  
--task-option ipa2_run_phasing=True \  
--task-option ipa2_run_purge_dups=True \  
--task-option ipa2_ctg_prefix="ctg." \  
--task-option ipa2_reads_db_prefix="reads" \  
--task-option ipa2_cleanup_intermediate_files=True \  
--task-option dataset_filters="" \  
--task-option filter_min_qv=20 \  
--nproc 8
```

The default options for this workflow should work well for any genome types.

If the assembly is run on a single local node with high CPU count, such as 64 cores, we recommend that the job submission for `pbcrumwell` is configured so that it uses 4 concurrent jobs and 16 threads per job.

We found this to be more efficient than using 64 threads and 1 concurrent job, as many steps are very data I/O-dependent.

You can apply a similar principle for compute environments with more or fewer cores. For example, for a machine with 80 cores, one can use 20 threads and 4 concurrent jobs.

Genome Assembly parameters input files

Option	Default value	Description
<code>-e, --eid_ccs</code>	NONE	Optional parameter, required if <code>--task-option reads</code> <code><input></code> is not specified. This is a SMRT Link-specific input parameter and supports only PacBio Consensusreadset XML files as input.
<code>--task-option reads</code>	NONE	Optional parameter, required if <code>-e <input></code> is not specified. Supports multiple input formats: FASTA, FASTQ, BAM, XML, FOFN and gzipped versions of FASTA/FASTQ.
<code>--task-option ipa2_genome_size</code>	0	The approximate number of base pairs expected in the genome. This is used only for downsampling; if the value is ≤ 0 , downsampling is disabled. Note: It is better to slightly overestimate rather than underestimate the genome length to ensure good coverage across the genome.
<code>--task-option ipa2_downsampled_coverage</code>	0	The input Data Set can be downsampled to a desired coverage, provided that both the <code>ipa2_downsampled_coverage</code> and <code>ipa2_genome_size</code> options are specified and >0 . Downsampling applies to the entire assembly process, including polishing. This parameter selects reads randomly, using a fixed random seed for reproducibility.
<code>--task-option ipa2_advanced_options</code>	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.)
<code>--task-option ipa2_run_polishing</code>	TRUE	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies.
<code>--task-option ipa2_run_phasing</code>	TRUE	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies.
<code>--task-option ipa2_run_purge_dups</code>	TRUE	Enables or disables identification of "duplicate" alternate haplotype contigs which may be assembled in the primary contig file, and moves them to the associate contig (haplotig) file.
<code>--task-option ipa2_ctg_prefix</code>	.ctg	The prefix used to label the output generated contigs.
<code>--task-option ipa2_reads_db_prefix</code>	reads	The prefix of the sequence and seed databases which will be used internally for assembly.
<code>--task-option ipa2_cleanup_intermediate_files</code>	TRUE	Removes intermediate files from the run directory to save space.
<code>--task-option dataset_filters</code>	NONE	(General pbcromwell option) A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
<code>--task-option filter_min_qv</code>	20	(General pbcromwell option) Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
<code>--task-option downsample_factor</code>	0	Downsampling factor applied directly to the input Data Set. This parameter is not related to <code>ipa2_downsampled_coverage</code> .
<code>--task-option mem_scale_factor</code>	8	Controls the amount of requested memory for individual Cromwell tasks in the workflow. The default value of 8 is good for larger genomes, but it may be too much memory for smaller genomes.

Option	Default value	Description
<code>--config</code>	NONE	(General pbcrumwell option) Java configuration file for running Cromwell.
<code>--nproc</code>	1	(General pbcrumwell option) Number of processors, (except per task).

- `*.bam` file containing PacBio data.
- `*.fasta` or `*.fastq` file containing PacBio data.
- `*.xml` file containing PacBio data.
- `*.fofn` files with file names of files containing PacBio data.

Output files

- `final_purged_primary.fasta` file containing assembled primary contigs.
- `final_purged_haplotigs.fasta` file containing assembled haplotigs.

Advanced parameters

Advanced parameters should be rarely modified. For the special cases when that is required, advanced parameters are documented below.

Advanced parameters specified on the command line:

- Are in the form of `key = value` pairs.
- Each pair is separated by a semicolon (`;`) character.
- The full set of advanced parameters is surrounded by **one** set of double quotes.
- The specified value of a parameter **overwrites** the default options for that key. **All** desired options of that parameter must be explicitly listed, not just the ones which should change from the default.
- Setting an empty value **clears** the parameter; it does **not** reset the value back to default.

Example

```
--task-option ipa2_advanced_options="config_seeddb_opt=-k 28;config_block_size=2048"
```

Complete list of available advanced parameters and default values:

Advanced parameters	Default value	Description
config_genome_size	0	The approximate number of base pairs expected in the genome, used to determine the coverage cutoff. This is only used for downsampling; 0 turns downsampling off. Note: It is better to slightly overestimate rather than underestimate the genome length to ensure good coverage across the genome.
config_coverage	0	The input Data Set can be downsampled to a desired coverage, provided that both the <code>Downsampled coverage</code> and <code>Genome Length</code> parameters are specified and above 0. Downsampling applies to the entire assembly process, including polishing. This feature selects reads randomly, using a fixed random seed for reproducibility.
config_polish_run	1	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies. 0 disables this feature; 1 enables it.
config_phase_run	1	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies. 0 disables this feature; 1 enables it.
config_purge_dups_run	1	Enables or disables the <code>purge_dups</code> stage of the workflow. 0 disables this feature; 1 enables it.
config_autocomp_max_cov	1	If enabled, the maximum allowed overlap coverage at either the 5' or the 3' end of every read is automatically determined based on the statistics computed from the overlap piles. This value is appended to the <code>config_ovl_filter_opt</code> value internally, and supersedes the manually specified <code>--max-cov</code> and <code>--max-diff</code> values of that parameter. These options are used to determine potential repeats and filter out those reads before the string graph is constructed. 0 disables this feature; 1 enables it.
config_block_size	4096	The overlapping process is performed on pairs of blocks of input sequences, where each block contains the number of sequences which crop up to this size (in Mbp). Note: The number of pairwise comparisons grows quadratically with the number of blocks (meaning more cluster jobs), but also the larger the block size the more resources are required to execute each pairwise comparison.
config_existing_db_prefix	NONE	Allows injection of an existing SeqDB, so that one doesn't have to be built from scratch. The provided existing DB is symbolically linked and used for assembly. (This option is intended for debugging purposes.)

Advanced parameters	Default value	Description
config_ovl_filter_opt	--max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 4000 --gapFilt --minDepth 4 --idt-stage2 98	<p>Overlap filter options.</p> <p>--gapFilt - Enables the chimera filter, which analyzes each overlap pile, and determines whether a pread is chimeric based on the local coverage across the pread.</p> <p>--minDepth - Option for the chimera filter. The chimera filter is ignored when a local region of a read has coverage lower than this value.</p> <p>The other parameters are:</p> <p>--min-cov - Minimum allowed coverage at either the 5' or the 3' end of a read. If the coverage is below this value, the read is blacklisted and all of the overlaps it is incident with are ignored. This helps remove potentially chimeric reads.</p> <p>--max-cov - Maximum allowed coverage at either the 5' or the 3' end of a read. If the coverage is above this value, the read is blacklisted and all of the overlaps it is incident with are ignored. This helps remove repetitive reads which can make tangles in the string graph. Note that this value is a heuristic which works well for ~30x seed length cutoff. If the cutoff is set higher, we advise that this value be also increased. Alternatively, using the <code>autocompute_max_cov</code> option can automatically estimate the value of this parameter, which can improve contiguity (for example, in cases when the input genome size or the seed coverage were overestimated).</p> <p>--max-diff - Maximum allowed difference between the coverages at the 5' and 3' ends of any particular read. If the coverage is above this value, the read is blacklisted and all of the overlaps it is incident with are ignored. If the <code>autocompute_max_cov</code> option is used, then the same computed value is supplied to this parameter as well.</p> <p>--bestn - Keep at most this many overlaps on the 5' and the 3' side of any particular read.</p> <p>--min-len - Filter overlaps where either A-read or the B-read are shorter than this value.</p> <p>--idt-stage2 - Filter overlaps with identity below 98%.</p> <p>--high-copy-sample-rate - Controls the downsampling of reads from high copy elements to the expected coverage determined by <code>maxCov*rate</code>, where <code>rate</code> is the value of this parameter. If <code>rate</code> is 0, then these high coverage reads are discarded.</p>
config_ovl_min_idt	98	The final overlap identity threshold. Applied during the final filtering stage, right before the overlaps are passed to the layout stage.
config_ovl_min_len	1000	The minimum length of either A-read or a B-read to keep the overlap. Applied during the final filtering stage, right before the overlaps are passed to the layout stage.
config_ovl_opt	--one-hit-per-target --min-idt 96	<p>Overlapping options for the <code>pancake</code> overlapping tool. The options set by this parameter here are passed directly to <code>pancake</code>. For details on <code>pancake</code> options, use <code>pancake -h</code>.</p> <p>The defaults used here are: <code>--one-hit-per-target</code> which keeps only the best hit in case there are multiple possible overlaps between a pair of reads (tandem repeats); and <code>--min-idt 96</code> which will filter out any overlap with identity lower than 96%.</p>
config_phasing_opt	NONE	Options for the phasing tool <code>nighthawk</code> . The options set by this parameter are passed directly to <code>nighthawk</code> . For details on <code>nighthawk</code> options, use <code>nighthawk -h</code> .

Advanced parameters	Default value	Description
config_phasing_split_opt	--split-type noverlaps --limit 3000000	Options that control the chunking of the phasing jobs, and through that regulate the time and memory consumption of each individual chunk. The defaults are: --split-type <code>noverlaps</code> which splits the chunks by the number of overlaps; and --limit 3000000 which allow at most approximately 3 million overlaps per chunk. Empirically, the current defaults keep the maximum memory consumption (RSS) of the phasing jobs under 4 GB per chunk.
config_seeddb_opt	-k 28 -w 120 --space 1	Options to control the seed computation. These options are passed directly to the <code>pancake seeddb</code> command. Defaults: -k 28 is the k-mer size of 28 bp; -w 120 is the minimizer window size of 120 bp; and --space 1 specifies the spacing for spaced seed construction, with 1 gap in between every two bases of the seed. For more details on these and other options, use <code>pancake seeddb -h</code> .
config_seqdb_opt	--compression 1	Options to control the construction of the sequence database. These options are passed directly to the <code>pancake seqdb</code> command. Current default is --compression 1 which turns on the 2-bit encoding compression of the sequences. For more details on these and other options, use <code>pancake seqdb -h</code> .
config_use_hpc	0	This parameter enables (1) or disables (0) an experimental Homopolymer Compression feature. If this feature is enabled, the overlaps are computed from homopolymer-compressed sequences. The layout stage is somewhat slower because the sequences have to be aligned to determine the correct homopolymer-expanded coordinates.
config_use_seq_ids	1	This feature is mostly useful for debugging purposes. If 0 is specified, then the overlaps contain original sequence names instead of their numerical IDs. The default of 1 uses the numerical IDs to represent reads, which uses memory much more efficiently.
config_purge_map_opt	--min-map-len 1000 --min-idt 98.0 --bestn 5	This option is used to control the mapping of the reads to contigs for the <code>purge_dups</code> tool. The mapper used is <code>pancake</code> , and the options set by this parameter are used directly by <code>pancake</code> . For details on <code>pancake</code> options, use <code>pancake -h</code> . Option --min-map-len 1000 removes any alignment which did not span more than 1000 bp during the mapping process; --min-idt 98.0 removes any alignment with identity below 98.0%, and --bestn 5 keeps at most 5 top scoring alignments for each query read (one primary alignment and at most 4 secondary alignments).

Advanced parameters	Default value	Description
<code>config_purge_dups_calcuts</code>	NONE	<p>The third-party tool <code>purge_dups</code> can accept user-defined cutoffs for purging. On some genomes, the automated computation of the cutoffs in <code>purge_dups</code> can result in suboptimal values, and in this case a user can specify them manually.</p> <p>This option is passed directly to the <code>purge_dups_calcuts</code> tool. For details on the possible values that can be passed to this tool, use <code>ipa_purge_dups_calcuts</code> without parameters.</p> <p>Relevant parameters include:</p> <ul style="list-style-type: none"> <code>-l</code> INT Lower bound for read depth. <code>-m</code> INT Transition between haploid and diploid. <code>-u</code> INT Upper bound for read depth.
<code>config_m4filt_high_copy_sample_rate</code>	1.0	<p>This option is passed to the <code>--high-copy-sample-rate</code> parameter of the overlap filter, which controls the downsampling of reads from high copy elements to the expected coverage determined by <code>maxCov*rate</code>, where <code>rate</code> is the value of this parameter. If 0, then these high coverage reads are discarded.</p> <p>Note: This parameter supersedes the <code>config_ovl_filter_opt</code> options.</p>
<code>config_max_polish_block_mb</code>	100	<p>During the polishing stage, contigs are grouped into chunks of approximate size specified by this parameter (in megabases). Each chunk is processed separately and in parallel (depending on the system configuration).</p>
<code>config_layout_opt</code>	NONE	<p>This value is passed directly to the assembly layout stage. To get a list of valid options for this parameter, enter <code>ipa2_ovlp_to_graph -h</code> on the command line.</p>

HiFiViral SARS-CoV-2 Analysis

Use this application to analyze multiplexed samples sequenced with the HiFiViral SARS-CoV-2 Kit. For **each** sample, this analysis provides:

- Consensus sequence (FASTA).
- Variant calls (VCF).
- HiFi reads aligned to the reference (BAM).
- Plot of HiFi read coverage depth across the SARS-CoV-2 genome.

Across **all** samples, this analysis provides:

- Job summary table including passing sample count at 90 and 95% genome coverage.
- Sample summary table including, for each sample: Count of variable sites, genome coverage, read coverage, and probability of multiple strains, and other metrics.
- Plate QC graphical summary of performance across samples in assay plate layout.
- Plot of HiFi read depth of coverage for all samples.

Notes:

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Phred-scaled Q20.
- This application is for SARS-CoV-2 analysis **only** and is **not** recommended for other viral studies. The Wuhan reference genome is included with SMRT Link and used by default, but advanced users may specify other reference genomes. We have **not** tested the application with reference genomes other than the Wuhan reference genome.
- The application is intended to identify variable sites and call a single consensus sequence per sample. The output consensus sequence is produced based on the dominant variant observed. Minor variant information that passes through a default threshold may be encoded in the raw VCF, but does **not** get propagated into the consensus sequence FASTA.
- The HiFiViral SARS-CoV-2 Analysis application can be run using the **Auto Analysis** feature available in Run Design. This feature allows users to complete all necessary analysis steps immediately after sequencing **without** manual intervention. The Auto Analysis workflow includes CCS, Demultiplex Barcodes, and HiFiViral SARS-CoV-2 Analysis.

HiFiViral SARS-CoV-2 application workflow

1. Process the reads using the `mimux` tool to trim the probe arm sequences.
2. Align the reads to the reference genome using `pbbmm2`.
3. Call and filter variants using `bcftools`, generating the raw variant calls in VCF file format. Filtering in this step removes low-quality calls (less than Q20), and normalizes indels.

4. Filter low-frequency variants using `vcfcons` and generate a consensus sequence by injecting variants into the reference genome. At each position, a variant is called only if both the base coverage exceeds the minimum base coverage threshold (Default = 4) and the fraction of reads that support this variant is above the minimum variant frequency threshold (Default = 0.5). See [here](#) for details.

Preparing input data for the HiFiViral SARS-CoV-2 Analysis application

1. Run the Demultiplex Barcodes `cromwell` workflow, where the input to that application are HiFi reads, and the primers are multiplexed barcode primers. See “[Demultiplex Barcodes](#)” on page 24 for details. If HiFi reads have **not** been generated on the instrument, run CCS analysis first. See “[ccs](#)” on page 8 for details.
 - Provide the proper barcode sequences:
Barcoded M13 Primer Plate.
 - Use the task option `lima_symmetric_barcodes=false`. (The barcode pairs are **asymmetric**.)
- Provide the correctly-formatted barcode pair-to-Bio Sample CSV file. For details, see the `--task-option sample_wells_csv` option in the table further down this page.

Input files

- `movie.consensusreadset.xml`: Previously-demultiplexed HiFi reads, packaged as separate BAM files wrapped in an XML Data Set. (See “[Preparing input data for the HiFiViral SARS-CoV-2 Analysis application](#)” on page 51 for details.)
- `sars_cov2.referenceset.xml`: The Wuhan reference genome.
- `HiFiViral_SARS-CoV-2_Enrichment_Probes.barcodeset.xml`: Dataset XML specifying the probe sequence file in FASTA format.
- **[Optional]** `Plate_QC_csv`: Four-column CSV file that includes `barcode`, `biosample`, `plateID` and `wellID`.
To specify this optional file, add the following option to `pbserve`:
`--task-option sample_wells_csv=<path to CSV file>`

Output files

- `pb_sars_cov2_kit.probe_counts_zip`: Zipped TSV files with probe counts, per sample.
- `pb_sars_cov2_kit.variants_csv`: CSV variant calls for **all** samples.
- `pb_sars_cov2_kit.vcf_zip`: Zipped VCF files containing the final variant calls, per sample.
- `pb_sars_cov2_kit.raw_vcf_zip`: Zipped VCF files containing the raw variant calls, per sample.
- `pb_sars_cov2_kit.fasta_zip`: Zipped final consensus sequences, by sample, in FASTA format. This is a single consensus sequence with `Ns` for each sample.
- `pb_sars_cov2_kit.frag_fasta_zip`: Zipped file of consensus sequences, split on `Ns`, in FASTA format, by sample.

- `pb_sars_cov2_kit.mapped_zip`: Zipped BAM files containing the output from mapping HiFi reads to the reference genome, by sample.
- `samples.consensus_mapped.bam.zip`: Zipped BAM files containing the output from mapping consensus FASTA files to the reference genome, by sample.
- `samples.coverage.png.zip`: Zipped per-sample coverage graphs in png format.
- `hifi_reads.fastq.zip`: Zipped file of per-sample trimmed HiFi reads in FASTQ format.
- `sample_summary.csv`: Sample Summary file in CSV format, with one row per sample.

Options	Description
<code>--task-option sample_wells_csv</code>	Specifies a 4-column CSV file used to generate the Plate QC Report, which displays analysis results for each sample in the assay plate. The CSV file must contain barcode pairs, Bio Sample name, Plate IDs, and Well IDs. The report is useful for diagnosing sample issues based on plate location. (Default = None)
<code>--task-option min_coverage</code>	Specifies the minimum read coverage. Below this value, the consensus sequence will be set to Ns and no variants are called. (Default = 4)
<code>--task-option min_alt_freq</code>	Specifies that only variants whose frequency is greater than this value are reported. This frequency is determined based on the read depth (DP) and allele read count (AD) information in the VCF output file. We recommend using the default value to properly call the dominant alternative variant while also filtering out potential artifacts. (Default = 0.5)
<code>--task-option min_bq</code>	Specifies that reads with barcode scores below this minimum value are not included in analysis. (Default = 80)
<code>--task-option mimux_overrides</code>	Specifies additional options to pass to the <code>mimux</code> preprocessing tool for trimming and filtering reads by probe sequences. Options should be entered in space-separated format. Available options include: <code>--max-len</code> : Specifies the maximum sequence length. (Default = 800) <code>--same</code> : Specifies that only reads with arms sequences from the same probe are used.
<code>--task-option probes_fasta</code>	Specifies a FASTA file containing probes sequences if using probes other than those supplied with the HiFiViral SARS-CoV-2 kit.

Running the SARS-CoV-2 Analysis application

```

pbcromwell run pb_sars_cov2_kit \

-e <movie.consensusreadset.xml> \

-e $SMRT_ROOT/current/bundles/smrtinub/current/private/pacbio/barcodes/HiFiViral_SARS-
CoV-2_Enrichment_Probes.barcodeset.xml \

-e eid_ref_dataset_2:$SMRT_ROOT/current/bundles/smrtinub/current/private/pacbio/
canneddata/referenceset/SARS-CoV-2/sars_cov2.referenceset.xml

--task-option min_alt_freq=0.5 \
--task-option min_bq=80 \
--task-option mimux_overrides="--max-len=800 --same" \
--task-option sample_wells_csv=None \
--config cromwell.conf \
--nproc 8

```

ipdSummary The `ipdSummary` tool detects DNA base-modifications from kinetic signatures. It is part of the `kineticsTool` package.

`kineticsTool` loads IPDs observed at each position in the genome, compares those IPDs to value expected for unmodified DNA, and outputs the result of this statistical test. The expected IPD value for unmodified DNA can come from either an in-silico control or an amplified control. The in-silico control is trained by PacBio and shipped with the package. It predicts the IPD using the local sequence context around the current position. An amplified control Data Set is generated by sequencing unmodified DNA with the same sequence as the test sample. An amplified control sample is usually generated by whole-genome amplification of the original sample.

Modification detection

The basic mode of `kineticsTool` does an independent comparison of IPDs at each position on the genome, for each strand, and outputs various statistics to CSV and GFF files (after applying a significance filter).

Modifications identification

`kineticsTool` also has a Modification Identification mode that can decode multi-site IPD “fingerprints” into a reduced set of calls of specific modifications. This feature has the following benefits:

- Different modifications occurring on the same base can be distinguished; for example, 6mA and 4mC.
- The signal from one modification is combined into one statistic, improving sensitivity, removing extra peaks, and correctly centering the call.

Algorithm: Synthetic control

Studies of the relationship between IPD and sequence context reveal that most of the variation in mean IPD across a genome can be predicted from a 12-base sequence context surrounding the active site of the DNA polymerase. The bounds of the relevant context window correspond to the window of DNA in contact with the polymerase, as seen in DNA/polymerase crystal structures. To simplify the process of finding DNA modifications with PacBio data, the tool includes a pre-trained lookup table mapping 12-mer DNA sequences to mean IPDs observed in C2 chemistry.

Algorithm: Filtering and trimming

`kineticsTool` uses the Mapping QV generated by `pbmm2` and stored in the `cmp.h5` or BAM file (or AlignmentSet) to **ignore** reads that are not confidently mapped. The default minimum Mapping QV required is 10, implying that `pbmm2` has 90% confidence that the read is correctly mapped. Because of the range of read lengths inherent in PacBio data, this can be changed using the `--mapQvThreshold` option.

There are a few features of PacBio data that require special attention to achieve good modification detection performance. `kineticsTool` inspects the alignment between the observed bases and the reference sequence for an IPD measurement to be included in the analysis. The PacBio read sequence **must** match the reference sequence for k around the cognate base. In the current module, $k=1$. The IPD distribution at some locus can be thought of as a mixture between the “normal” incorporation process IPD, which is sensitive to the local sequence context and DNA modifications, and a contaminating “pause” process IPD, which has a much longer duration (mean >10 times longer than normal), but happen rarely (~1% of IPDs).

Note: Our current understanding is that pauses do **not** carry useful information about the methylation state of the DNA; however a more careful analysis may be warranted. Also note that modifications that drastically increase the roughly 1% of observed IPDs are generated by pause events. Capping observed IPDs at the global 99th percentile is motivated by theory from robust hypothesis testing. Some sequence contexts may have naturally longer IPDs; to avoid capping too much data at those contexts, the cap threshold is adjusted per context as follows:

```
capThreshold = max(global99, 5*modelPrediction,
percentile(ipdObservations, 75))
```

Algorithm: Statistical testing

We test the hypothesis that IPDs observed at a particular locus in the sample have longer means than IPDs observed at the same locus in unmodified DNA. If we have generated a Whole Genome Amplified Data Set, which removes DNA modifications, we use a case-control, two-sample t-test. This tool also provides a pre-calibrated “synthetic control” model which predicts the unmodified IPD, given a 12-base sequence context. In the synthetic control case we use a one-sample t-test, with an adjustment to account for error in the synthetic control model.

Usage

To run using a BAM input, and output GFF and HDF5 files:

```
ipdSummary aligned.bam --reference ref.fasta m6A,m4C --gff basemods.gff \
--csv_h5 kinetics.h5
```

To run using `cmp.h5` input, perform methyl fraction calculation, and output GFF and CSV files:

```
ipdSummary aligned.cmp.h5 --reference ref.fasta m6A,m4C --methylFraction \
--gff basemods.gff --csv kinetics.csv
```

Output options	Description
<code>--gff FILENAME</code>	GFF format.
<code>--csv FILENAME</code>	Comma-separated value format.
<code>--bigwig FILENAME</code>	BigWig file format.

Input files

- A standard PacBio alignment file - either AlignmentSet XML, BAM, or `cmp.h5` - containing alignments and IPD information.
- Reference sequence used to perform alignments. This can be either a FASTA file or a ReferenceSet XML.

Output files

The tool provides results in a variety of formats suitable for in-depth statistical analysis, quick reference, and consumption by visualization tools. Results are generally indexed by reference position and reference strand. In all cases the strand value refers to the strand carrying the modification in the DNA sample. Remember that the kinetic effect of the modification is observed in read sequences aligning to the opposite strand. So reads aligning to the positive strand carry information about modification on the negative strand and vice versa, but the strand containing the putative modification is always reported.

- `modifications.gff`: Compliant with the GFF Version 3 [specification](#). Each template position/strand pair whose probability value exceeds the probability value threshold appears as a row. The template position is 1-based, per the GFF specifications. The strand column refers to the strand carrying the detected modification, which is the opposite strand from those used to detect the modification. The GFF confidence column is a Phred-transformed probability value of detection.

The auxiliary data column of the GFF file contains other statistics which may be useful for downstream analysis or filtering. These include the coverage level of the reads used to make the call, and +/- 20 bp sequence context surrounding the site.

- `modifications.csv`: Contains one row for each (reference position, strand) pair that appeared in the Data Set with coverage at least `x`. `x` defaults to 3, but is configurable with the `--minCoverage` option. The reference position index is 1-based for compatibility with the GFF file in the R environment. Note that this output type scales poorly and is **not** recommended for large genomes; the HDF5 output should perform much better in these cases.

Output columns: In-silico control mode

Column	Description
refId	Reference sequence ID of this observation.
tpl	1-based template position.
strand	Native sample strand where kinetics were generated. 0 is the strand of the original FASTA, 1 is opposite strand from FASTA.
base	The cognate base at this position in the reference.
score	Phred-transformed probability value that a kinetic deviation exists at this position.
tMean	Capped mean of normalized IPDs observed at this position.
tErr	Capped standard error of normalized IPDs observed at this position (standard deviation/sqrt(coverage)).
modelPrediction	Normalized mean IPD predicted by the synthetic control model for this sequence context.
ipdRatio	tMean/modelPrediction.
coverage	Count of valid IPDs at this position.
frac	Estimate of the fraction of molecules that carry the modification.
fracLow	2.5% confidence bound of the frac estimate.
fracUpp	97.5% confidence bound of the frac estimate.

Output columns: Case control mode

Column	Description
refId	Reference sequence ID of this observation.
tpl	1-based template position.
strand	Native sample strand where kinetics were generated. 0 is the strand of the original FASTA, 1 is opposite strand from FASTA.
base	The cognate base at this position in the reference.
score	Phred-transformed probability value that a kinetic deviation exists at this position.
caseMean	Mean of normalized case IPDs observed at this position.
controlMean	Mean of normalized control IPDs observed at this position.
caseStd	Standard deviation of case IPDs observed at this position.
controlStd	Standard deviation of control IPDs observed at this position.
ipdRatio	tMean/modelPrediction.
testStatistic	T-test statistic.
coverage	Mean of case and control coverage.
controlCoverage	Count of valid control IPDs at this position.
caseCoverage	Count of valid case IPDs at this position.

isoseq3 The `isoseq3` tool enables analysis and functional characterization of transcript isoforms for bulk and single-cell sequencing data generated on PacBio instruments. The analysis can be performed *de novo*, without a reference genome. If a reference genome is available, an optional collapse step to produce unique isoform files based on genomic coordinates is available. The input to `isoseq3` tools should be HiFi (CCS) reads in BAM format.

Recommended commands for bulk Iso-Seq® Analysis

Command	Description	Output format	Recommended next step
<code>lima</code>	Remove cDNA primers.	<code>fl.bam</code>	<code>isoseq3 refine</code>
<code>isoseq3 refine</code>	Remove polyA tail and artificial concatemers.	<code>flnc.bam</code>	<code>isoseq3 cluster</code>
<code>isoseq3 cluster</code>	<i>De novo</i> isoform-level clustering.	<code>unpolished.bam</code>	<code>pbbmm2</code>
<code>pbbmm2</code>	Align to the genome.	<code>aligned.sorted.bam</code>	<code>isoseq3 collapse</code>
<code>isoseq3 collapse</code>	Collapse redundant transcripts based on exonic structures.	<code>collapsed.gff</code>	<code>pigeon</code>

Recommended commands for single-cell Iso-Seq® Analysis

Command	Description	Output format	Recommended next step
<code>lima</code>	Remove cDNA primers.	<code>fl.bam</code>	<code>isoseq3 tag</code>
<code>isoseq3 tag</code>	Extract UMIs and cell barcodes.	<code>fl.tagged.bam</code>	<code>isoseq3 refine</code>
<code>isoseq3 refine</code>	Remove polyA tail and artificial concatemers.	<code>flnc.bam</code>	<code>isoseq3 correct</code>
<code>isoseq3 correct</code>	Correct cell barcodes and tag reads that are real cells.	<code>flnc.corrected.bam</code>	<code>isoseq3 bcstats</code>
<code>isoseq3 bcstats</code>	Summarize barcode statistics for real/non-real cells.	<code>bcstats.reports.tsv</code>	<code>isoseq3 dedup</code> or <code>isoseq3 groupdedup</code>
<code>isoseq3 dedup</code> or <code>isoseq3 groupdedup</code>	Deduplicate UMIs.	<code>dedup.bam</code>	<code>pbbmm2</code>
<code>pbbmm2</code>	Align to the genome.	<code>aligned.sorted.bam</code>	<code>isoseq3 collapse</code>
<code>isoseq3 collapse</code>	Collapse redundant transcripts based on exonic structures.	<code>collapsed.gff</code>	<code>pigeon</code>

Usage

```
isoseq3 <tool>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits

Typical bulk Iso-Seq workflow

1. Visualize primers, then remove primers and demultiplex:

```
cat primers.fasta
>5p
GCAATGAAGTCGCAGGGTTGGGG
>3p
GTACTCTGCGTTGATACCACTGCTT
```

```
lima movie.ccs.bam primers.fasta demux.bam --isoseq
```

See [“Demultiplex Barcodes” on page 24](#) for details on the `lima` tool.

2. Identify and remove polyA tails; also remove artificial concatemers. The output are full-length, non-concatemer (FLNC) reads:

```
isoseq3 refine demux.5p--3p.bam primers.fasta flnc.bam --require-polya
```

3. Cluster FLNC reads at the isoform levels to generate consensus transcript isoform sequences. This generates `unpolished.hq.bam` and `unpolished.hq.fasta.gz` files, which are the high-quality (HQ) transcripts that should be analyzed further.

```
isoseq3 cluster flnc.bam unpolished.bam --use-qvs
```

4. (Optional) Map transcripts to the genome and collapse HQ transcripts based on genomic mapping:

```
pbmm2 align unpolished.bam reference.fasta aligned.sorted.bam --preset ISOSEQ --sort
isoseq3 collapse --do-not-collapse-extra-5exons aligned.sorted.bam out.gff or
isoseq3 collapse --do-not-collapse-extra-5exons aligned.sorted.bam movie.ccs.bam
out.gff
```

See [“pbmm2” on page 89](#) for details.

`refine` command: Remove polyA and concatemers from full-length (FL) reads and generate full-length non-concatemer (FLNC) transcripts (FL to FLNC).

Usage

```
isoseq refine [options] <ccs.demux.bam|xml> <primer.fasta|xml> <flnc.bam|xml>
```

Inputs/outputs	Description
<code>ccs.demux.bam xml</code>	Input demultiplexed CCS reads BAM or ConsensusReadSet XML file. This is usually the output from running <code>lima</code> with the <code>--isoseq</code> option, such as <code>demux.5p--3p.bam</code> .
<code>primer.fasta xml</code>	Input primer FASTA or BarcodeSet XML file.
<code>flnc.bam xml</code>	Output FLNC BAM or ConsensusReadSet XML file.

Preprocessing	Description
<code>--min-polya-length</code>	Specifies the minimum poly(A) tail length. (Default = 20)
<code>--require-polya</code>	Requires reads to have a poly(A) tail and remove it.
<code>--min-rq</code>	Specifies the minimum CCS read quality. (Default = -1, deactivated)

Options	Description
<code>--help, -h</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--verbose, -v</code>	Sets the verbosity level.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = stderr)
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARN, TRACE, FATAL]. (Default = WARN)

`cluster` command: Cluster FLNC reads and generate transcripts.

Usage

```
isoseq3 cluster [options] input output
```

Example

```
isoseq3 cluster flnc.bam unpolished.bam --use-gvs
```

Custom BAM tags

`isoseq3 cluster` adds the following custom PacBio tags to the output BAM file:

- `ib`: Barcode summary: triplets delimited by semicolons, each triplet contains two barcode indices and the ZMW counts, delimited by commas. **Example**: 0,1,20;0,3,5
- `im`: ZMW names associated with this isoform.
- `is`: Number of ZMWs associated with this isoform.

Inputs/outputs	Description
input	<code>flnc.bam</code> file or <code>movie.consensusreadset.xml</code> file.
output	<code>unpolished.bam</code> file prefix (the actual output files will be <code>unpolished.hq.bam</code> and <code>unpolished.lq-bam</code>) or <code>unpolished.transcriptset.xml</code> file.

Options	Description
<code>--s1</code>	Specifies the number of seeds for minimizer-only clustering. (Default = 1000)
<code>--s2</code>	Specifies the number of seeds for DP clustering. (Default = 1000)

Options	Description
<code>--poa-cov</code>	Specifies the maximum number of CCS reads used for POA consensus. (Default = 10)
<code>--use-qvs</code>	Use CCS analysis Quality Values; sets <code>--poa-cov</code> to 100.
<code>--split-bam</code>	Splits BAM output files into a maximum of <code>N</code> files; 0 means no splitting. (Default = 0)
<code>--min-subreads-split</code>	Subread threshold for High-Quality/Low-Quality split; only works with <code>--use-qvs</code> . (Default = 7)
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
<code>-v, --verbose</code>	Uses verbose output.
<code>-j, --num-threads</code>	Specifies the number of threads to use; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = stdout)

summarize command: Create a .csv-format barcode overview from transcripts.

Usage

```
isoseq3 summarize [options] input output
```

Example

```
isoseq3 summarize unpolished.bam summary.csv
```

Inputs/outputs	Description
input	unpolished.bam file. (The output from isoseq3 cluster.)
output	summary.csv file.

Options	Description
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
<code>-v, --verbose</code>	Uses verbose output.
<code>--log-file</code>	Writes the log to file. (Default = stdout)

collapse command: Collapse transcripts based on genomic mapping.

Usage

```
isoseq3 collapse [options] <alignments.bam|xml> <ccs.bam|xml> <out.fastq>
```

Example - Bulk Iso-Seq

```
isoseq3 collapse --do-not-collapse-extra-5exons aligned.sorted.bam out.gff or
isoseq3 collapse --do-not-collapse-extra-5exons aligned.sorted.bam ccs.bam out.gff
```

Example - Single-Cell Iso-Seq

```
isoseq3 collapse aligned.sorted.bam out.gff or
isoseq3 collapse aligned.sorted.bam ccs.bam out.gff
```

Inputs/outputs	Description
alignments	Alignments mapping transcripts to the reference genome. (BAM or XML file).
ccs.bam	Optional input BAM file containing CCS reads.
out.gff	Collapsed transcripts in GFF format.

Options	Description
--min-aln-coverage	Ignores alignments with less than the Minimum Query Coverage. (Default = 0.95)
--min-aln-identity	Ignores alignments with less than the Minimum Alignment Identity. (Default = 0.50)
--max-fuzzy-junction	Ignores mismatches or indels shorter than or equal to N. (Default = 5)
--max-5p-diff	Specifies the maximum allowed 5' difference if on same exon. (Default = 1000)
--max-3p-diff	Specifies the maximum allowed 3' difference if on same exon. (Default = 100)
--do-not-collapse-extra-5exons	Do not collapse 5' shorter transcripts which miss one or multiple 5' exons to a longer transcript.
--max-batch-mem	Specifies the maximum memory for batch loading, in megabytes (MB). Batches can be slightly larger than this value. Value ≤ 0 loads all data in memory at once. (Default = 4096)
--split-group-size	Specifies that groups larger than this will be linearly split for parallel processing. (Default = 100)
--keep-non-real-cells	Do not skip reads with non-real cells.
--version	Displays program version number and exits.
--log-file	Writes the log to file. (Default = stderr)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, ERROR, CRITICAL]. (Default = WARN)
-j, --num-threads	Specifies the number of threads to use; 0 means autodetection. (Default = 0)

correct command: Correct group barcodes ($x_c:z$) given a set of known correct barcodes. Additionally, real (as opposed to background RNA) cells are annotated with the `rc` tag after this step. In subsequent steps, such as `isoseq3 groupdedup`, only reads from real cells as indicated by the `rc` tag will be used.

Usage

```
isoseq3 correct [options] <flnc.bam> <flnc.corrected.bam>
```

Inputs/outputs	Description
flnc.bam	Input BAM file containing full-length non-concatemer reads, tagged with UMIs and cell barcodes.
flnc.corrected.bam	Output BAM file containing barcode-corrected full-length non-concatemer reads.

Options	Description
--barcodes, -B	Specifies a plain text file containing known "true" barcodes, one per line. This include list specifies the barcode-set to which raw cell barcodes are remapped by minimum edit distance. May be gzip-compressed.
--max-edit-distance, -M	Specifies the maximum edit distance for mapping barcodes to those specified by the -barcodes option. Increasing this value increases yield, but potentially introduce errors. (Default = 2)
--filter, -F	Specifies the filtering mode. <ul style="list-style-type: none"> missing removes reads which could not be corrected. failing removes reads which could not be corrected as well as reads failing the -max-edit-distance threshold. (Default = none)
--help, -h	Displays help information and exits.
--version	Displays program version number and exits.
--verbose, -v	Sets the verbosity level.
-j, --num-threads	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
--log-file	Writes the log to a file. (Default = stderr)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, TRACE, FATAL]. (Default = WARN)

tag command: Remove cell barcodes and unique molecular identifiers (UMIs) from full-length reads and generate tagged full-length transcripts (FL to FLT).

Usage

```
isoseq3 tag [options] <fl.bam> <fl.tagged.bam>
```

Inputs/outputs	Description
fl.bam	Input BAM file containing full-length reads.
fl.tagged.bam	Output BAM file containing full-length reads, with UMI and cell barcodes extracted and tagged.

Options	Description
--design	Specifies which bases to use as cell/molecular barcodes as part of the barcoding design. (Default = T-8U-10B)
--min-read-length	Specifies the minimum read length after trimming. (Default = 50)
--help, -h	Displays help information and exits.
--version	Displays program version number and exits.

Options	Description
<code>--verbose, -v</code>	Sets the verbosity level.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are <code>[DEBUG, INFO, WARN, TRACE, FATAL]</code> . (Default = <code>WARN</code>)

`groupdedup` command: Deduplicate reads grouped by UMIs (barcode-sorted FLTNC to DEDUP). Note that after barcode correction, the BAM file **must** be sorted by the corrected barcode BAM tag (`CB`) first.

Usage

```
samtools sort -t CB corrected.bam -o corrected.sorted.bam
isoseq3 groupdedup [options] <fltnc.bam|xml (one or more).>
```

Inputs/outputs	Description
<code>fltnc.bam xml (one or more)</code>	Input BAM file(s) containing full-length non-concatemer transcripts, sorted by corrected barcode BAM tag (<code>CB</code>).
<code>dedup.bam</code>	Output BAM file(s) containing deduplicated reads.

Options	Description
<code>--poa-cov</code>	Specifies the maximum number of CCS reads used to determine <code>poa</code> consensus, using the <code>poa</code> variant-calling algorithm. (Default = 10)
<code>--max-tag-mismatches</code>	Specifies the maximum number of mismatches between tags. (Default = 1)
<code>--max-tag-shift</code>	Specifies that tags may be shifted by at maximum of <code>N</code> bases. (Default = 1)
<code>--max-length-difference</code>	Specifies the maximum insert lengths difference. (Default = 50)
<code>--max-insert-pad</code>	Specifies the maximum number of missing flanking bases on either insert side. (Default = 5)
<code>--min-concordance-perc</code>	Specifies the minimum insert alignment concordance, in %. (Default = 97)
<code>--max-insert-gaps</code>	Specifies the maximum number of insert gaps per 20 bp window. (Default = 5)
<code>--barcode-tag, -C</code>	Specifies the BAM cell/group tag by which to group. If not specified, checks the <code>CB</code> and <code>XC</code> tags.
<code>--no-poa</code>	Select the highest-pass read in the data set instead of generating a consensus sequence using the <code>poa</code> variant-calling algorithm.
<code>--keep-non-real-cells</code>	Specifies not to skip reads with non-real cells, as indicated by the <code>rc</code> tag.
<code>--batch-size</code>	Specifies the number of BAM records to load per batch. (Default = 500000)
<code>--help, -h</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.

Options	Description
<code>--verbose, -v</code>	Sets the verbosity level.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARN, TRACE, FATAL]. (Default = WARN)

`bcstats` command: Generates statistics for group barcodes (`CB:Z`, `XC:Z`, or `CR:Z`) and (optionally) molecular barcodes (`XM:Z`, `UR:Z`, or `UB:Z`). This tool should be run after `isoseq3 correct` to get the `rc` tag, which indicates which reads come from real cells (as opposed to ambient RNA).

Usage

```
isoseq3 bcstats [options] <Input BAMs>
```

Inputs	Description
<code>input.bam</code> (one or more)	Input BAM file(s) containing full-length non-concatemer transcripts.

Options	Description
<code>--molecular, -U</code>	Specifies statistics for molecular barcodes (UMIs) as well as cell barcodes; this produces a <code>.tsv</code> file with one line for each cell barcode and one line for each molecular barcode.
<code>-o, --output</code>	Specifies an output <code>.tsv</code> file of statistics for input BAM files. (Default = <code>/dev/stdout</code>)
<code>-R, --json</code>	Specifies the path to an output JSON report. (Default = <code>/dev/stderr</code>)
<code>--deduplicated, -D</code>	Specifies that the output is deduplicated. This allows for faster analysis and sanity checks across versions.
<code>--batch-size, -Z</code>	Specifies the batch size for processing. (Default = 65536)
<code>--percentile, -P</code>	Specifies the percentile to use when calculating real vs non-real cells. This option is relevant only when <code>RealCellCalculationMethod</code> (<code>--method</code>) is set to <code>percentile</code> . (Default = 99)
<code>-T, --target</code>	Specifies whether to determine real vs non-real cells by read count (<code>readcount</code>) or UMI count (<code>umicount</code>). (Default = <code>umicount</code>)
<code>-M, --method</code>	Specifies whether to determine real vs non-real cells using the Knee-finding (<code>knee</code>) or the percentile-based method (<code>percentile</code>). (Default = <code>knee</code>)
<code>--help, -h</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--verbose, -v</code>	Sets the verbosity level.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Writes the log to a file. (Default = <code>stderr</code>)
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARN, TRACE, FATAL]. (Default = WARN)

`dedup` command: Deduplicate reads grouped by UMIs. **Note:** This tool has been superseded by the `groupdedup` tool.

Usage

```
isoseq3 dedup [options] <fltnc.bam|xml> <dedup.bam|xml>
```

Inputs/outputs	Description
fltnc.bam xml (one or more)	Input BAM file(s) containing full-length non-concatemer transcripts, sorted by corrected barcode BAM tag (CB).
dedup.bam xml	Output BAM file(s) or XML file(s) containing deduplicated reads.

Options	Description
--poa-cov	Specifies the maximum number of CCS reads used to determine poa consensus, using the poa variant-calling algorithm. (Default = 10)
--max-tag-mismatches	Specifies the maximum number of mismatches between tags. (Default = 1)
--max-tag-shift	Specifies that tags may be shifted by at maximum of N bases. (Default = 1)
--max-insert-length-diff	Specifies the maximum insert lengths difference. (Default = 50)
--max-insert-pad	Specifies the maximum number of missing flanking bases on either insert side. (Default = 5)
--min-concordance-perc	Specifies the minimum insert alignment concordance, in %. (Default = 97)
--max-insert-gaps	Specifies the maximum number of insert gaps per 20 bp window. (Default = 5)
--keep-non-real-cells	Specifies not to skip reads with non-real cells, as indicated by the rc tag.
--help, -h	Displays help information and exits.
--version	Displays program version number and exits.
--verbose, -v	Sets the verbosity level.
-j, --num-threads	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
--log-file	Writes the log to a file. (Default = stderr)
--log-level	Specifies the log level; values are [DEBUG, INFO, WARN, TRACE, FATAL]. (Default = WARN)

juliet `juliet` is a general-purpose minor variant caller that identifies and phases minor single nucleotide substitution variants in complex populations. It identifies codon-wise variants in coding regions, performs a reference-guided *de novo* variant discovery, and annotates known drug-resistance mutations. Insertion and deletion variants are currently ignored; support will be added in a future version. There is no technical limitation with respect to the target organism or gene.

The underlying model is a statistical test, the Bonferroni-corrected Fisher's Exact test. It compares the number of observed mutated codons to the number of expected mutations at a given position.

`juliet` uses JSON target configuration files to define different genes in longer reference sequences, such as overlapping open reading frames in HIV. These predefined configurations ease batch applications and allow immediate reproducibility. A target configuration may contain multiple coding regions within one reference sequence and optional drug resistance mutation positions.

Notes:

- The preinstalled target configurations are meant for a quick start. It is the user's responsibility to ensure that the target configurations used are correct and up-to-date.
- If the target configuration `none` was specified, the provided reference is assumed to be in-frame.

Performance

At a coverage of 6,000 CCS reads with a predicted accuracy (RQ) of ≥ 0.99 , the false positive and false negative rates are below 1% and 0.001% (10^{-5}), respectively.

Usage

```
juliet --config "HIV" data.align.bam patientZero.html
```

Required	Description
<code>input_file.bam</code>	Input aligned BAM file containing CCS reads, which must be PacBio-compliant, that is, <code>cigar M</code> is forbidden.
<code>output_file.html</code>	Output report HTML file.

Configuration	Description
<code>--config, -c</code>	Path to the target configuration JSON file, predefined target configuration tag, or the JSON string.
<code>--mode-phasing, -p</code>	Phase variants and cluster haplotypes.

Restrictions	Description
<code>--region, -r</code>	Specifies the genomic region of interest; reads are clipped to that region. Empty means all reads.
<code>--drm-only, -k</code>	Only reports DRM positions specified in the target configuration. Can be used to filter for drug-resistance mutations - only known variants from the target configuration are called.
<code>--min-perc, -m</code>	Specifies the minimum variant percentage to report. Example: <code>--min-perc 1</code> will only show variant calls with an observed abundance of more than 1%. (Default = 0)
<code>--max-perc, -n</code>	Specifies the maximum variant percentage to report. Example: <code>--max-perc 95</code> will only show variant calls with an observed abundance of less than 95%. (Default = 100)

Chemistry override (specify both)	Description
<code>--sub, -s</code>	Specifies the substitution rate. Use to override the learned rate. (Default = 0)
<code>--del, -d</code>	Specifies the deletion rate. Use to override the learned rate. (Default = 0)

Options	Description
<code>--help, -h</code>	Displays help information and exits.
<code>--verbose, -v</code>	Sets the verbosity level.
<code>--version</code>	Displays program version number and exits.
<code>--debug</code>	Returns all amino acids, irrespective of their relevance.
<code>--mode-phasing, -p</code>	Phases variants and cluster haplotypes.

Input files

- BAM-format files containing CCS reads. These must be PacBio-compliant, that is, `cigar M` is forbidden.
- Input CCS reads should have a minimal predicted accuracy of 0.99.
- Reads should be created with CCS analysis using the `--richQVs` option. Without the `--richQVs` information, the number of false positive calls might be higher, as `juliet` is missing information to filter actual heteroduplexes in the sample provided.
- `juliet` currently does **not** demultiplex barcoded data; you must provide one BAM file per barcode.

Output files

A JSON and/or HTML file:

```
juliet data.align.bam patientZero.html
juliet data.align.bam patientZero.json
juliet data.align.bam patientZero.html patientZero.json
```

The HTML file includes the same content as the JSON file, but in more human-readable format. The HTML file contains four sections:

1. Input data

Summarizes the data provided, the exact call for `juliet`, and `juliet` version for traceability purposes.

2. Target config

Summarizes details of the provided target configuration for traceability. This includes the configuration version, reference name and length, and annotated genes. Each gene name (in bold) is followed by the reference start, end positions, and possibly known drug resistance mutations.

▼ Target config

Config Version: Predefined v1.1, PacBio internal

Reference Name: HIV_HXB2

Reference Length: 9719

Genes:

- **5'LTR** (1-634)
- **p17** (790-1186)
- **p24** (1186-1879)
- **p2** (1879-1921)
- **p7** (1921-2086)
- **p1** (2086-2134)
- **p6** (2134-2292)
- **Protease** (2253-2550)
 - ATV/r: V32I L33F M46I M46L I47V G48V G48M I50L I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88S L90M
 - DRV/r: V32I L33F I47V I47A I50V I54L I54M L76V V8F I84V
 - FPV/r: V32I L33F M46I M46L I47V I47A I50V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V L90M
 - IDV/r: V32I M46I M46L I47V I54V I54T I54A I54L I54M L76V V82A V82T V82F V82S I84V N88S L90M
 - NFV: D30N L33F M46I M46L I47V G48V G48M I54V I54T I54A I54L I54M V82A V82T V82F V82S I84V N88D N88S L90M
 - SQV/r: G48V G48M I54V I54T I54A I54L I54M V82A V82T I84V N88S L90M
 - TPV/r: V32I L33F M46I M46L I47V I47A I54V I54A I54M V82T V82L I84V

3. Variant discovery

For each gene/open reading frame, there is one overview table.

Each row represents a variant position.

- Each variant position consists of the reference codon, reference amino acid, relative amino acid position in the gene, mutated codon, percentage, mutated amino acid, coverage, and possible affected drugs.
- Clicking the row displays counts of the multiple-sequence alignment counts of the -3 to +3 context positions.

▼ Variant Discovery

HIV HXB2			Reverse Transcriptase						
Codon	AA	Pos	Sample Variants						
AA	Codon	%	Coverage	Affected Drugs*					
A T G	M	41	L	T T G	1	2793	ABC + DDI + TDF + D4T + ZDV		
A A A	K	65	R	A G A	1.1	2529	3TC + FTC + ABC + DDI + TDF + D4T		
			Pos	A	C	G	T	-	N
			-3	2947	0	0	0	0	51
			-2	2923	0	2	0	0	73
			-1	4	0	2952	0	0	42
			0	2606	0	0	0	339	53
			1	2905	0	29	0	0	64
			2	2938	0	0	0	0	60
			3	2938	0	0	0	0	60
			4	2942	0	0	0	0	56
			5	2751	0	0	0	0	247
T A T	Y	181	C	T G T	0.91	2946	NVP + EFV + ETR + RPV		
G G A	G	190	A	G C A	1	2947	NVP + EFV + ETR + RPV		
A C C	T	215	Y	T A C	0.93	2877	ABC + DDI + TDF + D4T + ZDV		

*HIVdb version 8.3 (last updated 2017-03-02)

► Legend

4. Drug summaries

Summarizes the variants grouped by annotated drug mutations:

▼ Drug Summaries

Drug	Gene	Reference AA	Pos	Sample AA	Sample %
3TC	Reverse Transcriptase	K	65	R	1
ABC	Reverse Transcriptase	M	41	L	0.99
		K	65	R	1
		T	215	Y	0.88

Predefined target configuration

juliet ships with one predefined target configuration, for HIV. Following is the command syntax for running that predefined target configuration:

```
juliet --config "HIV" data.align.bam patientZero.html
```

p6								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
A A C	N	47	S	A G T	0.95	2924		
Protease								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
C G A	R	8	X	T G A	0.98	2931		
Reverse Transcriptase								
HIV HXB2			Sample Variants					
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs [*]	
A T G	M	41	L	T T G	0.99	2903	ABC + DDI + TDF + D4T + ZDV	
A A A	K	65	R	A G A	1	2577	3TC + FTC + ABC + DDI + TDF + D4T	
T T A	L	100	F	T T T	0.85	2819		
T A T	Y	181	C	T G T	0.95	2939	NVP + EFV + ETR + RPV	
G G A	G	190	A	G C A	1	2941	NVP + EFV + ETR + RPV	
A C C	T	215	Y	T A C	0.88	2940	ABC + DDI + TDF + D4T + ZDV	

- **Note:** For the predefined configuration `HIV`, use the HIV HXB2 complete genome for alignment.

Customized target configuration

To define your own target configuration, create a JSON file. The root child `genes` contains a list of coding regions, with `begin` and `end`, the name of the gene, and a list of drug resistant mutations. Each DRM consists of its name and the positions it targets. The `drms` field is optional. If provided, the `referenceSequence` is used to call mutations, otherwise it will be tested against the major codon. All indices are with respect to the provided alignment space, 1-based, begin-inclusive and end-exclusive `[]`.

Target configuration: Example 1- A customized `json` target configuration file named `my_customized_hiv.json`:

```
{
  "genes": [
    {
      "begin": 2550,
      "drms": [
        {
          "name": "fancy drug",
          "positions": [ "M41L" ]
        }
      ],
      "end": 2700,
      "name": "Reverse Transcriptase"
    }
  ],
  "referenceName": "my seq",
  "referenceSequence": "TGGAAGGGCT..."
}
```

```

"version": "Free text to version your config files"
"databaseVersion": "DrugDB version x.y.z (last updated YYYY-MM-DD)"
}

```

Run with a customized target configuration using the `--config` option:

```
juliet --config my_customized_hiv.json data.align.bam patientZero.html
```

Valid formats for DRMs/positions

103	Only the reference position.
M130	Reference amino acid and reference position.
M103L	Reference aa, reference position, mutated aa.
M103LKA	Reference aa, reference position, list of possible mutated aas.
103L	Reference position and mutated aa.
103LG	Reference position and list mutated aas.

Missing amino acids are processed as wildcard (*).

Example

```
{ "name": "ATV/r", "positions": [ "V32I", "L33", "46IL",
  "I54VTALM", "V82ATFS", "84" ] }
```

Target configuration: Example 2 - BCR-ABL:

For BCR-ABL, using the ABL1 gene with the following [reference](#) NM_005157.5, a typical target configuration looks like this:

```

{
  "genes": [
    {
      "name": "ABL1",
      "begin": 193,
      "end": 3585,
      "drms": [
        {
          "name": "imatinib",
          "positions": [
            "T315AI", "Y253H", "E255KV", "V299L", "F317AICLV", "F359CIV" ]
        },
        {
          "name": "dasatinib",
          "positions": [ "T315AI", "V299L", "F317AICLV" ]
        },
        {
          "name": "nilotinib",
          "positions": [ "T315AI", "Y253H", "E255KV", "F359CIV" ]
        },
        {
          "name": "bosutinib",
          "positions": [ "T315AI" ]
        }
      ]
    }
  ],
  "referenceName": "NM_005157.5",
  "referenceSequence": "TTAACAGGCGCGTCCC..."
}

```

No target configuration

If **no** target configuration is specified, either make sure that the sequence is in-frame, or specify the region of interest to mark the correct reading frame, so that amino acids are correctly translated. The output is labeled with `unknown` as the gene name:

```
juliet data.align.bam patientZero.html
```

Phasing

The default mode is to call amino-acid/codon variants independently. Using the `--mode-phasing` option, variant calls from distinct haplotypes are clustered and visualized in the HTML output.

Protease										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
C G A	R	8	X	T G A	0.98	2931	MGI											

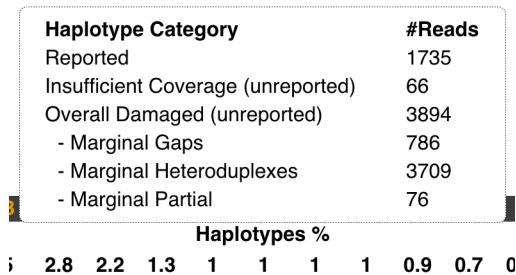
Reverse Transcriptase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A T G	M	41	L	T T G	0.99	2903	ABC + DDI + TDF + D4T + ZDV											
A A A	K	65	R	A G A	1	2577	3TC + FTC + ABC + DDI + TDF + D4T											
G G G	G	99	G	G G T	0.72	2907												
T T A	L	100	F	T T T	0.85	2819	MGI											
T A T	Y	181	C	T G T	0.95	2939	NVP + EFV + ETR + RPV											
G G A	G	190	A	G C A	1	2941	MGI + NVP + EFV + ETR + RPV											
A C C	T	215	Y	T A C	0.88	2940	ABC + DDI + TDF + D4T + ZDV											

Integrase										A	B	C	D	E	F	G	H	I
HXB2		Sample Variants								Haplotypes %								
Codon	AA	Pos	AA	Codon	%	Coverage	Affected Drugs*			92.5	1.2	1.2	1	1	0.8	0.8	0.8	0.7
A A A	K	188	K	A A G	0.92	2923	MGI											

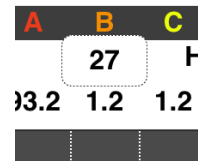
- The row-wise variant calls are "transposed" onto per-column haplotypes. Each haplotype has an ID: `[A-Z]{1}[a-z]?`.
- For each variant, colored boxes in this row mark haplotypes that contain this variant.
- Colored boxes per haplotype/column indicate variants that co-occur. Wild type (no variant) is represented by plain dark gray. A color palette helps to distinguish between columns.
- The JSON variant positions has an additional `haplotype_hit` boolean array with the length equal to the number of haplotypes. Each entry indicates if that variant is present in the haplotype. A haplotype block under the root of the JSON file contains counts and read names. The order of those haplotypes matches the order of all `haplotype_hit` arrays.

There are two types of tooltips in the haplotype section of the table.

The first tooltip is for the **Haplotypes %** and shows the number of reads that count towards (A) Actually reported haplotypes, (B) Haplotypes that have less than 10 reads and are not being reported, and (C) Haplotypes that are not suitable for phasing. Those first three categories are mutually exclusive and their sum is the total number of reads going into *juliet*. For (C), the three different marginals provide insights into the sample quality; as they are marginals, they are not exclusive and can overlap. The following image shows a sample with bad PCR conditions:



The second type of tooltip is for each haplotype percentage and shows the number of reads contributing to this haplotype:



**Microbial
Genome
Analysis**

The Microbial Genome Analysis application is powered by the IPA HiFi genome assembler and includes a base modification detection feature performed after the assembly.

Workflow of the Microbial Genome Analysis application

The workflow consists of several steps around 2 main stages:

1. **Chromosomal stage:** Assemble large contigs using IPA, the HiFi genome assembly tool.
2. Separate reads that were used for accurate large contigs from all other reads.
3. **Plasmid stage:** Assemble plasmids (using IPA) from the reads separated in the previous step.
4. De-duplicate plasmids.
5. Collect all contigs into a single FASTA file.
6. Rotate circular contigs.
7. Align the input Data Set to the assembled contigs.
8. Polish assembled contigs using `Racon`.
9. Perform base modification detection.

The application accepts HiFi XML Data Sets as input, and has an embedded downsampling feature:

- If the genome size and the desired coverage are specified, **both** stages of assembly are downsampled, as with the Genome Assembly application.
- Otherwise, the full coverage is used.

The embedded downsampling feature is **not** applied to the alignment stage; **all** input reads will be aligned against the assembled contigs.

Usage

The Microbial Genome Analysis application is run using the `pbcrumwell` run command, with the `pb_microbial_analysis` parameter to specify the application. See “[pbcrumwell](#)” on page 80 for details.

To view information on the available Microbial Genome Analysis options, enter:

```
pbcrumwell show-workflow-details pb_microbial_analysis
```

The minimum command needed to run the workflow requires the input Data Set.

The following example performs assembly and base modification detection using an input XML Data Set, and uses all default settings, including 1 CPU:

```
pbcrumwell run pb_microbial_analysis -e <input.consensusreadset.xml>
```

Note: To specify different task options on the command line, consider the following example:

```
pbccromwell run pb_microbial_analysis \
-e <input.consensusreadset.xml> \
--task-option ipa2_genome_size=0 \
--task-option ipa2_downsampled_coverage=0 \
--task-option microasm_plasmid_contig_len_max=300000
--task-option ipa2_cleanup_intermediate_files=True \
--task-option dataset_filters="" \
--task-option filter_min_qv=20 \
--nproc 8
```

The default options for this application should work well for any genome type.

As microbes are relatively small, we rarely find much advantage to using more than 4 threads, or more than 2 concurrent jobs.

Microbial Genome Analysis parameters

Option	Default value	Description
-e, --eid_ccs	NONE	This is a SMRT Link-specific input parameter and supports only PacBio Consensusreadset XML files as input.
--task-option reads	NONE	Optional parameter, required if -e <input> is not specified. Supports multiple input formats: FASTA, FASTQ, BAM, XML, FOFN and gzipped versions of FASTA/FASTQ.
--task-option ipa2_genome_size	10M	The approximate number of base-pairs expected in the chromosomal genome. Used only for downsampling in the assembly stages (that is, not in polishing). If value <=0, then downsampling is off. Default: 10M (10 Mega basepairs). Note: ipa2_genome_size is currently the only option that accepts a metric suffix. Commas and decimals are not accepted anywhere. ipa2_genome_size actually accepts a String, which can be an integer followed by one of these metric suffixes: k/M/G. For example: 4500k means "4,500 kilobases" or "4,500,000". M stands for Mega and G stands for Giga.
--task-option ipa2_downsampled_coverage	100	The maximum coverage after downsampling with respect to the estimated genome_size. (The default genome_size was chosen to be larger than most realistic microbes.) If value <=0, then downsampling is off. Default: 100 Example: If your genome is 1G in length, and you specify ipa2_genome_size=2G, you have over-estimated by 2x. If you also specify ipa2_downsampled_coverage=100, your data will be downsampled to 200x coverage, simply because of the over-estimate. The cost of extra coverage is greater runtime. The defaults are usually fine.

Option	Default value	Description
--task-option ipa2_advanced_options_chrom	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) config_block_size = 100; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75
--task-option ipa2_advanced_options_plasmid	See Description column	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. (These are described later in this document.) con-fig_block_size = 100; con-fig_ovl_filter_opt = --max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 500 --gapFilt --minDepth 4 --idt-stage2 98; con-fig_ovl_min_len = 500; con-fig_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --min-map-len 500 --min-anchor-span 500 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75 --smart-hit-per-target --secondary-min-ovl-frac 0.05; con-fig_layout_opt = -allow-circular;
--task-option ipa2_cleanup_intermediate_files	TRUE	Removes intermediate files from the run directory to save space.
--task-option microasm_plasmid_contig_len_max	300000	After the chromosomal stage, in task filter_draft_contigs, separates long contigs (presumed to be chromosomal) from shorter contigs (to be re-assembled in the plasmid stage). Then, after the plasmid stage, in dedup_plasmids, contigs less than this value are ignored. The default value is usually fine.
--task-option microasm_run_secondary_polish	TRUE	Specifies that an additional round of polishing will be applied after the two stages of assembly have completed.
--task-option run_basemods	TRUE	Specifies that base modification analysis be performed.
--task-option kineticstools_identify_mods	m4C,m6A	Specify the base modifications to identify, in a comma-separated list.
--task-option kineticstools_p_value	0.001	Specifies the probability value cutoff for detecting base modifications.
--task-option motif_min_score	35	Specifies the minimum QMod score used to identify a motif.
--task-option motif_min_fraction	0.30	Specifies the minimum methylated fraction to identify a motif.
--task-optionrun_find_motifs	TRUE	Specifies that motif-finding be performed.
--task-option dataset_filters	NONE	(General pbcromwell option) A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
--task-option filter_min_qv	20	(General pbcromwell option) Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.

Option	Default value	Description
<code>--task-option log_level</code>	INFO	Specifies the logging (verbosity) level for tools which support this feature. Values are: [TRACE, DEBUG, INFO, WARN, FATAL]
<code>--task-option nproc</code>	1	Specified the number of CPU threads to use.
<code>--task-option max_nchunks</code>	40	Specifies the maximum number of chunks per task.
<code>--task-option tmp_dir</code>	/tmp	Specifies an optional temporary directory for used by several subtools, such as <code>sort</code> .

Input files

- `*.bam` file containing PacBio data.
- `*.xml` file containing PacBio data.

Output files

- `final_assembly.fasta`: File containing all assembled contigs, rotated.
- `assembly.rotated.polished.renamed.fsa`: File for NCBI, but otherwise identical to `final_assembly.fasta`; only the headers are changed.
- `pb_microbial_analysis.motifs_csv`: File containing motifs and modifications.

Advanced assembly options

Note: Advanced parameters should be rarely modified. Advanced parameters for the Microbial Genome Assembly workflow are the **same** as those for the Genome Assembly workflow. See [“Advanced parameters” on page 45](#) for details.

Advanced parameters specified on the command line:

- Are in the form of `key = value` pairs.
- Each pair is separated by a semicolon (;) character.
- The full set of advanced parameters is surrounded by **one** set of double quotes.
- The specified value of a parameter **overwrites** the default options for that key. **All** desired options of that parameter must be explicitly listed, not just the ones which should change from the default.
- Setting an empty value **clears** the parameter; it does **not** reset the value back to default.

Example

```
--task-option ipa2_advanced_options_chrom="config_seeddb_opt=-k
28;config_block_size=2048"
```

motifMaker The `motifMaker` tool identifies motifs associated with DNA modifications in prokaryotic genomes. Modified DNA in prokaryotes commonly arises from restriction-modification systems that methylate a specific base in a specific sequence motif. The canonical example is the m6A methylation of adenine in GATC contexts in *E. coli*. Prokaryotes may have a very large number of active restriction-modification systems present, leading to a complicated mixture of sequence motifs.

PacBio SMRT sequencing is sensitive to the presence of methylated DNA at single base resolution, via shifts in the polymerase kinetics observed in the real-time sequencing traces. For more background on modification detection, see [here](#).

Algorithm

Existing motif-finding algorithms such as MEME-chip and YMF are sub-optimal for this case for the following reasons:

- They search for a **single** motif, rather than attempting to identify a complicated mixture of motifs.
- They generally don't accept the notion of aligned motifs - the input to the tools is a window into the reference sequence which can contain the motif at any offset, rather than a single center position that is available with kinetic modification detection.
- Implementations generally either use a Markov model of the reference (MEME-chip), or do exact counting on the reference, but place restrictions on the size and complexity of the motifs that can be discovered.

Following is a rough overview of the algorithm used by `motifMaker`: Define a motif as a set of tuples: (position relative to methylation, required base). Positions not listed in the motif are implicitly degenerate. Given a list of modification detections and a genome sequence, define the following objective function on motifs:

$$\text{Motif score}(\text{motif}) = (\# \text{ of detections matching motif}) / (\# \text{ of genome sites matching motif}) * (\text{Sum of log-pvalue of detections matching motif}) = (\text{fraction methylated}) * (\text{sum of log-pvalues of matches})$$

Then, search (close to exhaustively) through the space of all possible motifs, progressively testing longer motifs using a branch-and-bound search. The “fraction methylated” term must be less than 1, so the maximum achievable score of a child node is the sum of scores of modification hits in the current node, allowing pruning of all search paths whose maximum achievable score is less than the best score discovered so far.

Usage

Run the `find` command, and pass the reference FASTA and the `modifications.gff.gz` file output by the PacBio modification detection workflow.

The `reprocess` subcommand annotates the GFF file with motif information for better genome browsing.

```
MotifMaker [options] [command] [command options]
```

`find` command: Run motif-finding.

```
find [options]
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>* -f, --fasta</code>	Reference FASTA file.
<code>* -g, --gff</code>	Modifications.gff or .gff.gz file.
<code>-m, --minScore</code>	Specifies the minimum Qmod score to use in motif finding. (Default = 40.0)
<code>* -o, --output</code>	Outputs motifs.csv file.
<code>-x, --xml</code>	Outputs motifs XML file.

`reprocess` command: Update a `modifications.gff` file with motif information based on new Modification QV thresholds.

```
reprocess [options]
```

Options	Description
<code>-c, --csv</code>	Raw modifications.csv file.
<code>* -f, --fasta</code>	Reference FASTA file.
<code>* -g, --gff</code>	Modifications.gff or .gff.gz file.
<code>-m, --minFraction</code>	Specifies that only motifs above this methylated fraction are used. (Default = 0.75)
<code>-m, --motifs</code>	Motifs.csv file.
<code>* -o, --output</code>	Reprocessed modifications.gff file.

Output files

Using the `find` command:

- **Output CSV file:** Contains motif information.

Using the `reprocess` command:

- **Output GFF file:** Contains added motif information based on new Modification QV thresholds. The format of the output file is the same as the input file.

pbccromwell The `pbccromwell` tool is PacBio's wrapper for the `cromwell` scientific workflow engine used to power SMRT Link. `pbccromwell` includes advanced utilities for interacting directly with a Cromwell server.

`pbccromwell` is designed primarily for running workflows distributed and supported by PacBio, but it is written to handle any valid WDL source (version 1.0), and is very flexible in how it takes input. PacBio workflows are expected to be found in the directory defined by the `SMRT_PIPELINE_BUNDLE_DIR` environment variable, which is automatically defined by the SMRT Link distribution.

Note that `pbccromwell` does **not** interact with SMRT Link services; to run a Cromwell workflow as a SMRT Link job, use `pbservice`. (For details, see ["pbservice" on page 96.](#))

Note: Interaction with the Cromwell server is primarily intended for developers and power users.

Usage

```
pbccromwell run [-h] [--output-dir OUTPUT_DIR] [--overwrite] [-i INPUTS]
                [-e ENTRY_POINTS] [-n NPROC] [-c MAX_NCHUNKS]
                [--target-size TARGET_SIZE] [--queue QUEUE] [-o OPTIONS]
                [-t TASK_OPTIONS] [-b BACKEND] [-r MAX_RETRIES]
                [--tmp-dir TMP_DIR] [--config CONFIG] [--dry-run]
                [run, show-workflows, show-workflow-details, configure, submit, get-
                job, abort, metadata, show-running, wait]
```

Options	Description
<code>--output-dir OUTPUT_DIR</code>	Specifies the output directory for <code>cromwell</code> output. (Default = <code>cromwell_out</code>)
<code>--overwrite</code>	Overwrites the output directory, if it exists. (Default = <code>False</code>)
<code>-i INPUTS</code> , <code>--inputs INPUTS</code>	Specifies <code>cromwell</code> inputs and settings as JSON files. (Default = <code>None</code>)
<code>-e ENTRY_POINTS</code> , <code>--entry ENTRY_POINTS</code>	Specifies the entry point Data Set; may be repeated for workflows that take more than one input Data Set. Note that all PacBio workflows require at least one such entry point.
<code>-n NPROC</code> , <code>--nproc NPROC</code>	Specifies the number of processors per task. (Default = <code>1</code>)
<code>-c MAX_NCHUNKS</code> , <code>--max-nchunks MAX_NCHUNKS</code>	Specifies the maximum number of chunks per task. (Default = <code>None</code>)
<code>--target-size TARGET_SIZE</code>	Specifies the target chunk size. (Default = <code>None</code>)
<code>--queue QUEUE</code>	Specifies the cluster queue to use. (Default = <code>None</code>)
<code>-o OPTIONS</code> , <code>--options OPTIONS</code>	Specifies additional <code>cromwell</code> engine options, as a JSON file. (Default = <code>None</code>)
<code>-t TASK_OPTIONS</code> , <code>--task-option TASK_OPTIONS</code>	Specifies workflow- or task-level option as <code>key=value</code> strings, specific to the application. May be specified multiple times for multiple options. (Default = <code>[]</code>)
<code>-b BACKEND</code> , <code>--backend BACKEND</code>	Specifies the backend to use for running tasks. (Default = <code>None</code>)
<code>-r MAX_RETRIES</code> , <code>--maxRetries MAX_RETRIES</code>	Specifies the maximum number of times to retry a failing task. (Default = <code>1</code>)

Options	Description
<code>--tmp-dir TMP_DIR</code>	Specifies an optional temporary directory for <code>cromwell</code> tasks, which must exist on all compute hosts. (Default = None)
<code>--config CONFIG</code>	Specifies a Java configuration file for running <code>cromwell</code> . (Default = None)
<code>--dry-run</code>	Specifies that <code>cromwell</code> is not executed, but instead writes out final inputs and then exits. (Default = True)
<code>workflow</code>	Specifies the workflow ID (such as <code>pb_ccs</code> or <code>cromwell.workflows.pb_ccs</code> for PacBio workflows only) or a path to a Workflow Description Language (WDL) source file.

Enter `pbccromwell {command} -h` for a command's options.

Examples

Show available PacBio-developed workflows:

```
pbccromwell show-workflows
```

Show details for a PacBio workflow:

```
pbccromwell show-workflow-details pb_ccs
```

Generate `cromwell.conf` with HPC settings:

```
pbccromwell configure --default-backend SGE --output-file cromwell.conf
```

Launch a PacBio workflow:

```
pbccromwell run pb_ccs -e /path/to/movie.subreadset.xml --nproc 8 --config /full/path/to/cromwell.conf
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. (Default = None, writes to stdout.)
<code>--log-level=INFO</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = INFO)
<code>--debug</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = None)

`pbccromwell run` command: Run a Cromwell workflow. Multiple input modes are supported, including a `pbsmrtpipe`-like set of arguments (for PacBio workflows **only**), and JSON files already in the native Cromwell format.

All PacBio workflows have similar requirements to the `pbsmrtpipe` pipelines in previous SMRT Link versions:

1. One or more PacBio dataset XML entry points, usually a SubreadSet or ConsensusReadSet (`--entry-point <FILE>.`)
2. Any number of workflow-specific task options (`--task-option <OPTION>.`)
3. Engine options independent of the workflow, such as number of processors per task (`--nproc`), or compute backend (`--backend`).

Output is directed to a new directory: `--output-dir`, which defaults to `cromwell_out`. This includes final inputs for the Cromwell CLI, and subdirectories for logs (workflow and task outputs), links to output files, and the Cromwell execution itself, which has a complex nested directory structure. Detailed information about the workflow execution can be found in the file `metadata.json`, in the native Cromwell format.

Note that output file links do **not** include the individual resource files of Data Sets and reports (BAM files, index files, plot PNGs, and so on.) Follow the symbolic links to their real path (for example using `readlink -f`) to find report plots.

For further information about Cromwell, see the official documentation [here](#).

Workflow examples

Run the CCS analysis:

```
pbccromwell run pb_ccs -e <SUBREADS> --nproc 8 --config /full/path/to/cromwell.conf
```

Run the Iso-Seq workflow, including mapping to a reference, and execute on SGE:

```
pbccromwell run pb_isoseq3 -e <SUBREADS> -e <PRIMERS> -e <REFERENCE> --nproc 8 -- config /full/path/to/cromwell.conf
```

Run a user-defined workflow:

```
pbccromwell run my_workflow.wdl -i inputs.json -o options.json --config /full/path/to/cromwell.conf
```

Set up input files but exit before starting Cromwell:

```
pbccromwell run pb_ccs -e <SUBREADS> --nproc 8 --dry-run
```

Print details about the named PacBio workflow, including input files and task options. **Note:** The prefix `cromwell.workflows.` is optional.

```
pbccromwell show-workflow-details pb_ccs
pbccromwell show-workflow-details cromwell.workflows.pb_ccs
```

`pbccromwell show-workflow-details` command: Display details about a specified PacBio workflow, including input files and task options.

Usage

```
pbccromwell show-workflow-details [-h] [--inputs-json INPUTS_JSON]
                                workflow_id
```

Options	Description
<code>workflow_id</code>	Specifies the workflow ID, such as <code>pb_ccs</code> or <code>cromwell.workflows.pb_ccs</code> for PacBio workflows only .
<code>--inputs-json INPUTS_JSON</code>	Writes a JSON template containing workflow inputs to the specified file. (Default = None)

`pbccromwell configure` command: Generate the Java configuration file used by `cromwell` to define backends and other important engine options that **cannot** be set at runtime. You can pass this to `pbccromwell run` using the `--config` argument.

Usage

```
pbccromwell configure [-h] [--port PORT]
                      [--local-job-limit LOCAL_JOB_LIMIT]
                      [--jms-job-limit JMS_JOB_LIMIT]
                      [--db-port DB_PORT] [--db-user DB_USER]
                      [--db-password DB_PASSWORD]
                      [--default-backend DEFAULT_BACKEND]
                      [--max-workflows MAX_WORKFLOWS]
                      [--output-file OUTPUT_FILE] [--timeout TIMEOUT]
                      [--cache] [--no-cache]
```

Options	Description
<code>--port PORT</code>	Specifies the port that <code>cromwell</code> should listen to. (Default = 8000)
<code>--local-job-limit LOCAL_JOB_LIMIT</code>	Specifies the maximum number of local jobs/tasks that can be run at once. (Default = 10)
<code>--jms-job-limit JMS_JOB_LIMIT</code>	Specifies the maximum number of jobs/tasks that can be submitted to the queueing system at one time. (Default = 500)
<code>--db-port DB_PORT</code>	Specifies the database port for <code>cromwell</code> to use; if undefined, database configuration is omitted. (Default = None)
<code>--db-user DB_USER</code>	Specifies the user name used to connect to Postgres. (Default = <code>smrtlink_user</code>)
<code>--db-password DB_PASSWORD</code>	Specifies the password used to connect to Postgres.
<code>--default-backend DEFAULT_BACKEND</code>	Specifies the default job execution backend. Choices are: Local, SGE, Slurm, PBS, or LSF. (Default = Local)
<code>--max-workflows MAX_WORKFLOWS</code>	Specifies the maximum number of workflows that <code>cromwell</code> can run at once. (Default = 100)
<code>--output-file OUTPUT_FILE</code>	Specifies the name of the output configuration file. (Default = <code>cromwell.conf</code>)
<code>--timeout TIMEOUT</code>	Specifies the time to wait for task completion before checking the cluster job status. (Default = 600)

Options	Description
--cache	Enables call caching. (Default = True)
--no-cache	Disables call caching. (Default = True)

pbindex The `pbindex` tool creates an index file that enables random access to PacBio-specific data in BAM files.

Usage

```
pbindex <input>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.

Input file

- `*.bam` file containing PacBio data.

Output file

- `*.pbi` index file, with the same prefix as the input file name.

pbmarkdup The `pbmarkdup` tool marks PCR duplicates in CCS reads Data Sets from amplified libraries. PCR duplicates are different reads that arose from amplifying a single-source molecule. `pbmarkdup` can also optionally remove the duplicate reads.

Note: `pbmarkdup` only works with CCS reads, **not** with Subreads.

`pbmarkdup` uses a reference-free comparison method. Duplicates are identified as pairs of reads that:

1. Have the same length - within 10 bp, and
2. Have high percent identity alignments at the molecule ends at >98% identity of the first and last 250 bp.

Clusters are formed from sets of two or more duplicate reads, and a single read is selected as the representative of each cluster. Other reads in the cluster are considered duplicates.

How are duplicates marked?

In FASTA and FASTQ formats, reads from duplicate clusters have annotated names. The following is a FASTA example:

```
>m64013_191117_050515/67104/ccs DUPLICATE=m64013_191117_050515/3802014/ccs DS=2
```

This shows a marked duplicate read `m64013_191117_050515/67104/ccs` that is a duplicate of `m64013_191117_050515/3802014/ccs` in a cluster with 2 reads (`DS` value). Accordingly, the following is the read selected as the representative of the molecule:

```
>m64013_191117_050515/3802014/ccs DS=2
```

In BAM format, duplicate reads are flagged with `0x400`. The read-level tag `ds` provides the number of reads in a cluster (like the `DS` attribute described above for FASTA/FASTQ), and the `du` tag provides the name of the representative read (like the `DUPLICATE` attribute described above for FASTA/FASTQ).

Usage

```
pbmarkdup [options] <INFILE.bam|xml|fa|fq|fofn> <OUTFILE.bam|xml|fa.gz|fq.gz>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file</code>	Logs to a file, instead of <code>stderr</code> .

Options	Description																
<code>--log-level</code>	<p>Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL] (Default = WARN)</p> <p><code>--log-level INFO</code> produces a summary report such as:</p> <table><tr><th>LIBRARY</th><th>READS</th><th>UNIQUE MOLECULES</th><th>DUPLICATE READS</th></tr><tr><td><Unnamed></td><td>25000</td><td>24948 (99.8%)</td><td>52 (0.2%)</td></tr><tr><td>SS-lib</td><td>496</td><td>493 (99.4%)</td><td>3 (0.6%)</td></tr><tr><td>TOTAL</td><td>25496</td><td>25441 (99.8%)</td><td>55 (0.2%)</td></tr></table>	LIBRARY	READS	UNIQUE MOLECULES	DUPLICATE READS	<Unnamed>	25000	24948 (99.8%)	52 (0.2%)	SS-lib	496	493 (99.4%)	3 (0.6%)	TOTAL	25496	25441 (99.8%)	55 (0.2%)
LIBRARY	READS	UNIQUE MOLECULES	DUPLICATE READS														
<Unnamed>	25000	24948 (99.8%)	52 (0.2%)														
SS-lib	496	493 (99.4%)	3 (0.6%)														
TOTAL	25496	25441 (99.8%)	55 (0.2%)														
<code>-j, --num-threads</code>	<p>Specifies the number of threads to use, 0 means autodetection. (Default = 0)</p>																

Duplicate marking options	Description
<code>--cross-library, -x</code>	Identifies duplicate reads across sequencing libraries. Libraries are specified in the BAM read group LB tag.

Output options	Description
<code>-rmdup, -r</code>	Excludes duplicates from OUTFILE. (This is redundant when <code>--dup-file</code> is specified.)
<code>--dup-file FILE</code>	Stores duplicate reads in an extra file other than OUTFILE. The format of this file can be different from the output file.
<code>--clobber, -f</code>	Overwrites OUTFILE if it exists.

Input files

CCS reads from one or multiple movies in any of the following formats:

- PacBio BAM (.ccs.bam)
- PacBio dataset (.dataset.xml)
- File of File Names (.fofn)
- FASTA (.fasta, .fasta.gz)
- FASTQ (.fastq, .fastq.gz)

Output files

CCS reads with duplicates marked in a format inferred from the file extension:

- PacBio BAM (.ccs.bam)
- PacBio dataset (.dataset.xml), which also generates a corresponding BAM file.
- FASTA (.fasta.gz)
- FASTQ (.fastq.gz)

Allowed input/output combinations

Input File	Output BAM	Output Dataset	Output FASTQ	Output FASTA
Input BAM	Allowed	Allowed	Allowed	Allowed
Input Dataset	Allowed	Allowed	Allowed	Allowed
Input FASTQ	Not Allowed	Not Allowed	Allowed	Allowed
Input FASTA	Not Allowed	Not Allowed	Not Allowed	Allowed

Examples

Run on a single movie:

```
pbmarkdup movie.ccs.bam output.bam
```

Run on multiple movies:

```
pbmarkdup movie1.fasta movie2.fasta output.fasta
```

Run on multiple movies and output duplicates in separate file:

```
pbmarkdup movie1.ccs.bam movie2.fastq uniq.fastq --dup-file dups.fasta
```

pbmm2 The `pbmm2` tool aligns native PacBio data, outputs PacBio BAM files, and is a SMRT `minimap2` wrapper for PacBio data.

`pbmm2` supports native PacBio input and output, provides sets of recommended parameters, generates sorted output on-the-fly, and post-processes alignments. Sorted output can be used directly for polishing using `GenomicConsensus`, if BAM has been used as input to `pbmm2`.

`pbmm2` adds the following SAM tag to each aligned record:

- `mg`, stores gap-compressed alignment identity, defined as $nM / (nM + nMM + nInsEvents + nDelEvents)$.

Usage

```
pbmm2 <tool>
```

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.

`index` command: Indexes references and stores them as `.mmi` files. Indexing is optional, but recommended if you use the same reference with the same `--preset` multiple times.

Usage

```
pbmm2 index [options] <ref.fa|xml> <out.mmi>
```

Input file

- `*.fasta`, `*.fa` file containing reference contigs or `*.referenceset.xml`.

Output file

- `out.mmi` (`minimap2` index file.)

Notes:

- You can use existing `minimap2 .mmi` files with `pbmm2 align`.
- If you use an index file, you **cannot** override parameters `-k`, `-w`, nor `-u` in `pbmm2 align`.
- The `minimap2` parameter `-H` (homopolymer-compressed k-mer) is always on for SUBREAD and UNROLLED presets, and can be disabled using `-u`.

Options	Description
<code>--preset</code>	Specifies the alignment mode: <ul style="list-style-type: none"> • "SUBREAD" -k 19 -w 10 • "CCS" -k 19 -w 10 -u • "ISOSEQ" -k 15 -w 5 -u • "UNROLLED" -k 15 -w 15 The option is not case-sensitive. (Default = SUBREAD)
<code>-k</code>	Specifies the k-mer size, which cannot be larger than 28. (Default = -1)
<code>-w</code>	Specifies the Minimizer window size. (Default = -1)
<code>-u, --no-kmer-compression</code>	Disables homopolymer-compressed k-mer. (Compression is on by default for the SUBREAD and UNROLLED presets.)
<code>--short-sa-cigar</code>	Populates the SA tag with the short cigar representation.

`align` command: Aligns PacBio reads to reference sequences. The output argument is optional; if not provided, the BAM output is streamed to stdout.

Usage

```
pbmm2 align [options] <ref.fa|xml|mml> <in.bam|xml|fa|fq> [out.aligned.bam|xml]
```

Input files

- *.fasta file containing reference contigs, or *.referenceset.xml, or *.mml index file.
- *.bam, *.subreadset.xml, *.consensusreadset.xml, *.transcriptset.xml, *.fasta, *.fa, *.fastq, or *.fastq file containing PacBio data.

Output files

- *.bam aligned reads in BAM format.
- *.alignmentset, *.consensusalignmentset.xml, or *.transcriptalignmentset.xml if XML output was chosen.

The following Data Set Input/output combinations are allowed:

SubreadSet > AlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.subreadset.xml hg38.movie.alignmentset.xml
```

ConsensusReadSet > ConsensusAlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.consensusreadset.xml
hg38.movie.consensusalignmentset.xml --preset CCS
```

TranscriptSet > TranscriptAlignmentSet

```
pbmm2 align hg38.referenceset.xml movie.transcriptset.xml
hg38.movie.transcriptalignmentset.xml --preset ISOSEQ
```

FASTA/FASTQ input

In addition to native PacBio BAM input, reads can also be provided in FASTA and FASTQ formats.

Attention: The resulting output BAM file **cannot** be used as input into GenomicConsensus!

With FASTA/FASTQ input, the `--rg` option sets the read group. **Example:**

```
pbmm2 align hg38.fasta movie.Q20.fastq hg38.movie.bam --preset CCS --rg
 '@RG\tID:myid\tSM:mysample'
```

All three reference file formats `.fasta`, `.referenceset.xml`, and `.mmi` can be combined with FASTA/FASTQ input.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--chunk-size</code>	Processes <i>N</i> records per chunk. (Default = 100)
<code>--sort</code>	Generates a sorted BAM file.
<code>-m, --sort-memory</code>	Specifies the memory per thread for sorting. (Default = 768M)
<code>-j, --alignment-threads</code>	Specifies the number of threads used for alignment. 0 means autodetection. (Default = 0)
<code>-J, --sort-threads</code>	Specifies the number of threads used for sorting. 0 means 25% of <code>-j</code> , with a maximum of 8. (Default = 0)
<code>--sample</code>	Specifies the sample name for all read groups. Defaults, in order of precedence: A) SM field in the input read group B) Biosample name C) Well sample name D) "UnnamedSample".
<code>--rg</code>	Specifies the read group header line such as <code>'@RG\tID:xyz\tSM:abc'</code> . Only for FASTA/Q inputs.
<code>-y, --min-gap-comp-id-perc</code>	Specifies the minimum gap-compressed sequence identity, in percent. (Default = 70)
<code>-l, --min-length</code>	Specifies the minimum mapped read length, in base pairs. (Default = 50)
<code>-N, --best-n</code>	Specifies the output at maximum <i>N</i> alignments for each read. 0 means no maximum. (Default = 0)
<code>--strip</code>	Removes all kinetic and extra QV tags. The output cannot be polished.
<code>--split-by-sample</code>	Specifies one output BAM file per sample.
<code>--no-bai</code>	Omits BAI index file generation for sorted output.
<code>--unmapped</code>	Specifies that unmapped records be included in the output.
<code>--median-filter</code>	Picks one read per ZMW of median length.
<code>--zmw</code>	Processes ZMW Reads; <code>subreadset.xml</code> input is required. This activates the UNROLLED preset.

Options	Description
<code>--hqregion</code>	Processes the HQ region of each ZMW; <code>subreadset.xml</code> input is required. This activates the UNROLLED preset.

Parameter set options and overrides	Description
<code>--preset</code>	Specifies the alignment mode: <ul style="list-style-type: none"> • "SUBREAD" <code>-k 19 -w 10 -o 5 -O 56 -e 4 -E 1 -A 2 -B 5 -z 400 -Z 50 -r 2000 -L 0.5</code> • "CCS" <code>-k 19 -w 10 -u -o 5 -O 56 -e 4 -E 1 -A 2 -B 5 -z 400 -Z 50 -r 2000 -L 0.5</code> • "ISOSEQ" <code>-k 15 -w 5 -u -o 2 -O 32 -e 1 -E 0 -A 1 -B 2 -z 200 -Z 100 -C 5 -r 200000 -G 200000 -L 0.5</code> • "UNROLLED" <code>-k 15 -w 15 -o 2 -O 32 -e 1 -E 0 -A 1 -B 2 -z 200 -Z 100 -r 2000 -L 0.5</code> (Default = SUBREAD)
<code>-k</code>	Specifies the k-mer size, which cannot be larger than 28. (Default = -1)
<code>-w</code>	Specifies the Minimizer window size. (Default = -1)
<code>-u, --no-kmer-compression</code>	Disables homopolymer-compressed k-mer. (Compression is on by default for the SUBREAD and UNROLLED presets.)
<code>-A</code>	Specifies the matching score. (Default = -1)
<code>-B</code>	Specifies the mismatch penalty. (Default = -1)
<code>-z</code>	Specifies the Z-drop score. (Default = -1)
<code>-Z</code>	Specifies the Z-drop inversion score. (Default = -1)
<code>-r</code>	Specifies the bandwidth used in chaining and DP-based alignment. (Default = -1)
<code>-o, --gap-open-1</code>	Specifies the gap open penalty 1. (Default = -1)
<code>-O, --gap-open-2</code>	Specifies the gap open penalty 2. (Default = -1)
<code>-e, --gap-extend-1</code>	Specifies the gap extension penalty 1. (Default = -1)
<code>-E, --gap-extend-2</code>	Specifies the gap extension penalty 2. (Default = -1)
<code>-L, --lj-min-ratio</code>	Specifies the long join flank ratio. (Default = -1)
<code>-G</code>	Specifies the maximum intron length; this changes <code>-r</code> . (Default = -1)
<code>-C</code>	Specifies the cost for a non-canonical GT-AG splicing. (Default = -1)
<code>--no-splice-flank</code>	Specifies that you do not prefer splicing flanks GT-AG.

Examples

Generate an index file for reference and reuse it to align reads:

```
pbmm2 index ref.fasta ref.mmi
pbmm2 align ref.mmi movie.subreads.bam ref.movie.bam
```

Align reads and sort on-the-fly, with 4 alignment and 2 sort threads:

```
pbmm2 align ref.fasta movie.subreads.bam ref.movie.bam --sort -j 4 -J 2
```

Align reads, sort on-the-fly, and create a PBI:

```
pbmm2 align ref.fasta movie.subreadset.xml ref.movie.alignmentset.xml --sort
```

Omit the output file and stream the BAM output to `stdout`:

```
pbmm2 align hg38.mmi movie1.subreadset.xml | samtools sort > hg38.movie1.sorted.bam
```

Align the CCS reads fastq input and sort the output:

```
pbmm2 align ref.fasta movie.Q20.fastq ref.movie.bam --preset CCS --sort --rg
 '@RG\tID:myid\tSM:mysample'
```

Alignment parallelization

The number of alignment threads can be specified using the `-j`, `--alignment-threads` option. If **not** specified, the maximum number of threads are used, minus one thread for BAM I/O and minus the number of threads specified for sorting.

Sorting

Sorted output can be generated using the `--sort` option.

- By default, 25% of threads specified with the `-j` option (Maximum = 8) are used for sorting.
- To override the default percentage, the `-J`, `--sort-threads` option defines the explicit number of threads used for on-the-fly sorting. The memory allocated per sort thread is defined using the `-m`, `--sort-memory` option, accepting suffixes M,G.

Benchmarks on human data show that 4 sort threads are recommended, but that no more than 8 threads can be effectively leveraged, even with 70 cores used for alignment. We recommend that you provide more memory to **each** of a **few** sort threads to avoid disk I/O pressure, rather than providing less memory to each of many sort threads.

What are parameter sets and how can I override them?

Per default, `pbmm2` uses recommended parameter sets to simplify the multitudes of possible combinations. See the available parameter sets in the option table shown earlier.

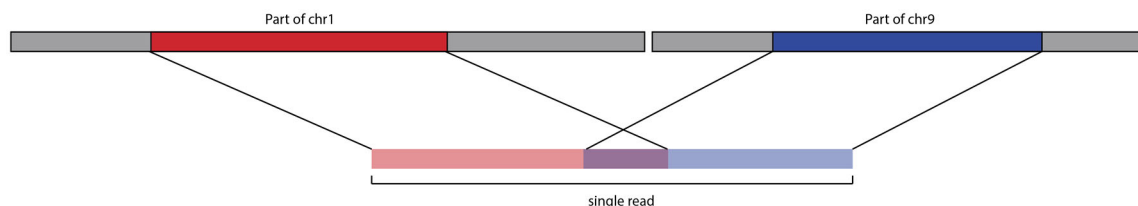
What other special parameters are used implicitly?

We implicitly use the following `minimap2` parameters:

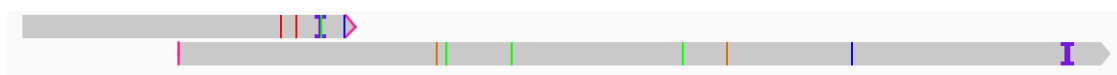
- Soft clipping with `-Y`.
- Long cigars for tag CG with `-L`.
- `X/=` cigars instead of `M` with `--eqx`.
- No overlapping query intervals with repeated matches trimming.
- No secondary alignments are produced using the `--secondary=no` option.

What is repeated matches trimming?

A repeated match occurs when the same query interval is shared between a primary and supplementary alignment. This can happen for translocations, where breakends share the same flanking sequence:



And sometimes, when a LINE gets inserted, the flanks are duplicated, leading to complicated alignments, where we see a split read sharing a duplication. The inserted region itself, mapping to a random other LINE in the reference genome, may also share sequence similarity to the flanks:



To get the best alignments, `minimap2` decides that two alignments may use up to 50% (default) of the same query bases. This does **not** work for PacBio, as `pbbmm2` requires that a single base may never be aligned twice. `Minimap2` offers a feature to enforce a query interval overlap to 0%. If a query interval gets used in two alignments, one or both get flagged as secondary and get filtered. This leads to yield loss, and more importantly, missing SVs in the alignment.

Papers (such as [this](#)) present dynamic programming approaches to find the optimal split to uniquely map query intervals, while maximizing alignment scores. We don't have per base alignment scores available, thus our approach is much simpler. We align the read, find overlapping query intervals, determine one alignment to be maximal reference-spanning, then trim all others. By trimming, `pbbmm2` rewrites the cigar and the reference coordinates on-the-fly. This allows us to increase the number of mapped bases, which slightly reduces mapped concordance, but boosts SV recall rate.

How can I set the sample name?

You can override the sample name (`SM` field in the `RG` tag) for **all** read groups using the `--sample` option. If not provided, sample names derive from the Data Set input using the following order of precedence: A) `SM` field in the input read group B) Biosample name C) Well sample name D) `UnnamedSample`. If the input is a BAM file and the `--sample` option was **not** used, the `SM` field is populated with `UnnamedSample`.

Can I split output by sample name?

Yes, the `--split-by-sample` option generates one output BAM file per sample name, with the sample name as the file name prefix, if there is more than one aligned sample name.

Can I remove all those extra per-base and per-pulse tags?

Yes, the `--strip` option removes the following extraneous tags if the input is BAM: `dq`, `dt`, `ip`, `iq`, `mq`, `pa`, `pc`, `pd`, `pe`, `pg`, `pm`, `pq`, `pt`, `pv`, `pw`, `px`, `sf`, `sq`, `st`. Note that the resulting output BAM file **cannot** be used as input into `GenomicConsensus`.

Where are the unmapped reads?

Per default, unmapped reads are omitted. You can add them to the output BAM file using the `--unmapped` option.

Can I output at maximum the N best alignments per read?

Use the option `-N`, `--best-n`. If set to 0, (the default), maximum filtering is disabled.

Is there a way to only align one subread per ZMW?

Using the `--median-filter` option, only the subread closest to the median subread length per ZMW is aligned. Preferably, full-length subreads flanked by adapters are chosen.

pbsservice The `pbsservice` tool performs a variety of useful tasks within SMRT Link.

- To get help for `pbsservice`, use `pbsservice -h`.
- To get help for a specific `pbsservice` command, use `pbsservice <command> -h`.

Note: `pbsservice` requires authentication when run from a remote host, using the same credentials used to log in to the SMRT Link GUI. (This also routes all requests through HTTPS port 8243, so the services port is **not** required if authentication is used.) Access to services running on `localhost` will continue to work without authentication.

All `pbsservice` commands include the following optional parameters:

Options	Description
<code>--host=http://localhost</code>	Specifies the server host. Override the default with the environmental variable <code>PB_SERVICE_HOST</code> .
<code>--port=8070</code>	Specifies the server port. Override the default with the environmental variable <code>PB_SERVICE_PORT</code> .
<code>--log-file LOG_FILE</code>	Writes the log to file. (Default = None, writes to stdout.)
<code>--log-level=INFO</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = INFO)
<code>--debug=False</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--quiet=False</code>	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
<code>--user USERNAME</code>	Specifies the user to authenticate as; this is required if the target host is anything other than <code>localhost</code> .
<code>--ask-pass</code>	Prompts the user to enter a password.
<code>--password PASSWORD</code>	Supplies the password directly. This exposes the password in the shell history (or log files), so this option is not recommended unless you are using a limited account without Unix login access.

`status` command: Use to get system status.

```
pbsservice status [-h] [--host HOST] [--port PORT]
                  [--log-file LOG_FILE]
                  [--log-level INFO]
                  [--debug] [--quiet]
```

`import-dataset` command: Import Local Data Set XML. The file location **must** be accessible from the host where the services are running; often on a shared file system.

```
pbsservice import-dataset [-h] [--host HOST] [--port PORT]
                           [--log-file LOG_FILE]
                           [--log-level INFO]
                           [--debug] [--quiet]
                           xml_or_dir
```

Required	Description
xml_or_dir	Specifies a directory or XML file for the Data Set.

import-run command: Create a SMRT Link Run Design and optional Auto Analysis jobs from a CSV file. This is equivalent to the Run Design CSV import feature in the SMRT Link UI; the resulting runs can then be edited in the UI or executed on-instrument.

```
pbservice import-run [-h][--host HOST] [--port PORT]
                    [--log-file LOG_FILE]
                    [--log-level INFO]
                    [--debug] [--quiet]
                    [--dry-run]
                    csv_file
```

Required	Description
csv_file	Specifies a Run Design CSV-format file.

Option	Description
--dry-run	Prints the generated run XML without POSTing to the server.

import-fasta command: Import a FASTA file and convert to a ReferenceSet file. The file location **must** be accessible from the host where the services are running; often on a shared file system.

```
pbservice import-fasta [-h] --name NAME --organism ORGANISM --ploidy
PLOIDY [--block] [--host HOST] [--port PORT]
      [--log-file LOG_FILE]
      [--log-level INFO]
      [--debug] [--quiet]
      fasta_path
```

Required	Description
fasta_path	Path to the FASTA file to import.

Options	Description
--name	Specifies the name of the ReferenceSet to convert the FASTA file to.
--organism	Specifies the name of the organism.
--ploidy	Ploidy.
--block=False	Blocks during importing process.

run-analysis command: Run a secondary analysis pipeline using an analysis.json file.

```
pbservice run-analysis [-h] [--host HOST] [--port PORT]
                      [--log-file LOG_FILE]
                      [--log-level INFO]
```

```

[--debug] [--quiet] [--block]
json_path

```

Required	Description
json_path	Path to the analysis.json file.

Options	Description
--block=False	Blocks during importing process.

emit-analysis-template command: Output an analysis.json template to stdout that can be run using the run-analysis command.

```

pbservice emit-analysis-template [-h] [--log-file LOG_FILE]
                                [--log-level INFO]
                                [--debug] [--quiet]

```

get-job command: Get a job summary by Job Id.

```

pbservice get-job [-h] [--host HOST] [--port PORT]
                  [--log-file LOG_FILE]
                  [--log-level INFO]
                  [--debug] [--quiet]
                  job_id

```

Required	Description
job_id	Job id or UUID.

get-jobs command: Get job summaries by Job Id.

```

pbservice get-jobs [-h] [-m MAX_ITEMS] [--host HOST] [--port PORT]
                  [--log-file LOG_FILE]
                  [--log-level INFO]
                  [--debug] [--quiet]

```

Options	Description
-m=25, --max-items=25	Specifies the maximum number of jobs to get.

get-dataset command: Get a Data Set summary by Data Set Id or UUID.

```

pbservice get-dataset [-h] [--host HOST] [--port PORT]
                      [--log-file LOG_FILE]
                      [--log-level INFO]
                      [--debug] [--quiet]
                      id_or_uuid

```

Required	Description
id_or_uuid	Data Set Id or UUID.

get-datasets command: Get a Data Set list summary by Data Set type.

```

pbservice get-datasets [-h] [--host HOST] [--port PORT]
                       [--log-file LOG_FILE]
                       [--log-level INFO]
                       [--debug] [--quiet] [-m MAX_ITEMS]

```

[-t DATASET_TYPE]

Required	Description
-t=subreads, --dataset-type=subreads	Specifies the type of Data Set to retrieve: subreads, alignments, references, barcodes.

delete-dataset command: Delete a specified Data Set.

Note: This is a "soft" delete - the database record is tagged as inactive so it won't display in any lists, but the files will **not** be removed.

```
pbsservice delete-dataset [-h] [--host HOST] [--port PORT]
                        [--log-file LOG_FILE]
                        [--log-level INFO]
                        [--debug] [--quiet]
                        [ID]
```

Required	Description
ID	A valid Data Set ID, either UUID or integer ID, for the Data Set to delete.

Examples

To obtain system status, the Data Set summary, and the job summary:

```
pbsservice status --host smrtlink-release --port 9091
```

To import a Data Set XML:

```
pbsservice import-dataset --host smrtlink-release --port 9091 \
path/to/subreadset.xml
```

To obtain a job summary using the Job Id:

```
pbsservice get-job --host smrtlink-release --port 9091 \
--log-level CRITICAL 1
```

To obtain Data Sets by using the Data Set type subreads:

```
pbsservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t subreads
```

To obtain Data Sets by using the Data Set type alignments:

```
pbsservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t alignments
```

To obtain Data Sets by using the Data Set type references:

```
pbsservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t references
```

To obtain Data Sets by using the Data Set type barcodes:

```
pbsservice get-datasets --host smrtlink-alpha --port 8081 \
--quiet --max-items 1 -t barcodes
```


To obtain Data Sets by using the Data Set UUID:

```
pbsservice get-dataset --host smrtlink-alpha --port 8081 \  
--quiet 43156b3a-3974-4ddb-2548-bb0ec95270ee
```

pbsv `pbsv` is a structural variant caller for PacBio reads. It identifies structural variants and large indels (Default: ≥ 20 bp) in a sample or set of samples relative to a reference. `pbsv` identifies the following types of variants: Insertions, deletions, duplications, copy number variants, inversions, and translocations.

`pbsv` takes as input read alignments (BAM) and a reference genome (FASTA); it outputs structural variant calls (VCF).

Usage

```
pbsv [-h] [--version] [--quiet] [--verbose]
      {discover,call}...
```

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of stdout.
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)
discover	Finds structural variant signatures in read alignments (BAM to SVSIG).
call	Calls structural variants from SV signatures and assign genotypes (SVSIG to VCF).

`pbsv discover` command: Finds structural variant (SV) signatures in read alignments. The input read alignments must be sorted by chromosome position. Alignments are typically generated with `pbbmm2`. The output SVSIG file contains SV signatures.

Usage

```
pbsv discover [options] <ref.in.bam|xml> <ref.out.svsig.gz>
```

Required	Description
ref.in.bam xml	Coordinate-sorted aligned reads in which to identify SV signatures.
ref.out.svsig.gz	Structural variant signatures output.

Options	Description
-h, --help	Displays help information and exits.
-s, --sample	Overrides sample name tag from BAM read group.
-q, --min-mapq	Ignores alignments with mapping quality < N. (Default = 20)
-m, --min-ref-span	Ignores alignments with reference length < N bp. (Default = 100)
-w, --downsample-window-length	Specifies a window in which to limit coverage, in base pairs. (Default = 10K)
-a, --downsample-max-alignments	Considers up to N alignments in a window; 0 means disabled. (Default = 100)

Options	Description
<code>-r, --region</code>	Limits discovery to this reference region: <code>CHR CHR:START-END</code> .
<code>-l, --min-sv-sig-length</code>	Ignores SV signatures with length $< N$ bp. (Default = 7)
<code>-b, --tandem-repeats</code>	Specifies tandem repeat intervals for indel clustering, as an input BED file.
<code>-k, --max-skip-split</code>	Ignores alignment pairs separated by $> N$ bp of a read or reference. (Default = 100)
<code>-y, --min-gap-comp-id-perc</code>	Ignores alignments with gap-compressed sequence identity $< N\%$. (Default = 70)

`pbsv call` command: Calls structural variants from SV signatures and assigns genotypes. The input SVSIG file is generated using `pbsv discover`. The output is structural variants in VCF format.

Usage

```
pbsv call [options] <ref.fa|xml> <ref.in.svsig.gz|fofn>
<ref.out.vcf>
```

Required	Description
<code>ref.fa xml</code>	Reference FASTA file or ReferenceSet XML file against which to call variants.
<code>ref.in.svsig.gz fofn</code>	SV signatures from one or more samples. This can be either an SV signature SVSIG file generated by <code>pbsv discover</code> , or a FOFN of SVSIG files.
<code>ref.out.vcf</code>	Variant call format (VCF) output file.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>-j, --num-threads</code>	Specifies the number of threads to use, 0 means autodetection. (Default = 0)
<code>-r, --region</code>	Limits discovery to this reference region: <code>CHR CHR:START-END</code>
<code>-t, --types</code>	Calls these SV types: "DEL", "INS", "INV", "DUP", "BND", "CNV". (Default = "DEL, INS, INV, DUP, BND")
<code>-m, --min-sv-length</code>	Ignores variants with length $< N$ bp. (Default = 20)
<code>--max-ins-length</code>	Ignores insertions with length $> N$ bp. (Default = 15K)
<code>--max-dup-length</code>	Ignores duplications with length $> N$ bp. (Default = 1M)
<code>--cluster-max-length-perc-diff</code>	Does not cluster signatures with difference in length $> P\%$. (Default = 25)
<code>--cluster-max-ref-pos-diff</code>	Does not cluster signatures $> N$ bp apart in the reference. (Default = 200)
<code>--cluster-min-basepair-perc-id</code>	Does not cluster signatures with base pair identity $< P\%$. (Default = 10)
<code>-x, --max-consensus-coverage</code>	Limits to N reads for variant consensus. (Default = 20)
<code>-s, --poa-scores</code>	Scores POA alignment with triplet <code>match, mismatch, gap</code> . (Default = "1, -2, -2")
<code>--min-realign-length</code>	Considers segments with $> N$ length for realignment. (Default = 100)

Options	Description
-A, --call-min-reads-all-samples	Ignores calls supported by < N reads total across samples. (Default = 3)
-O, --call-min-reads-one-sample	Ignores calls supported by < N reads in every sample. (Default = 3)
-S, --call-min-reads-per-strand-all-samples	Ignores calls supported by < N reads per strand total across samples. (Default = 1)
-B, --call-min-bnd-reads-all-samples	Ignores BND calls supported by < N reads total across samples. (Default = 2)
-P, --call-min-read-perc-one-sample	Ignores calls supported by < P% of reads in every sample. (Default = 20)
--preserve-non-acgt	Preserves non-ACGT in REF allele instead of replacing with N. (Default = false)
--hifi, --ccs	Uses options optimized for HiFi reads: -S 0 -P 10. (Default = False)
--gt-min-reads	Specifies the minimum supporting reads to assign a sample a non-reference genotype. (Default = 1)
--annotations	Annotates variants by comparing with sequences in FASTA. (Default annotations are ALU, L1, and SVA.)
--annotation-min-perc-sim	Annotates variant if sequence similarity >P%. (Default = 60)
--min-N-in-gap	Considers ≥ N consecutive "N" bp as a reference gap. (Default = 50)
--filter-near-reference-gap	Flags variants < N bp from a gap as "NearReferenceGap". (Default = 1000)
--filter-near-contig-end	Flags variants < N bp from a contig end as "NearContigEnd". (Default = 1K)

Following is a typical SV analysis workflow starting from subreads:

1. Align PacBio reads to a reference genome, per movie:

Subreads BAM input:

```
pbmm2 align ref.fa movie1.subreads.bam ref.movie1.bam --sort --median-filter --sample sample1
```

CCS reads BAM input:

```
pbmm2 align ref.fa movie1.ccs.bam ref.movie1.bam --sort --preset CCS --sample sample1
```

CCS reads FASTQ input:

```
pbmm2 align ref.fa movie1.Q20.fastq ref.movie1.bam --sort --preset CCS --sample sample1 --rg '@RG\tID:movie1'
```

2. Discover the signatures of structural variation, per movie or per sample:

```
pbsv discover ref.movie1.bam ref.sample1.svsig.gz
pbsv discover ref.movie2.bam ref.sample2.svsig.gz
```

3. Call structural variants and assign genotypes (all samples); for CCS analysis input append --ccs:

```
pbsv call ref.fa ref.sample1.svsig.gz ref.sample2.svsig.gz
ref.var.vcf
```

Launching a multi-sample pbsv analysis - requirements

1. Merge multiple Bio Sample SMRT Cells to one Data Set with the Bio Samples specified.
 - Each SMRT Cell must have exactly **one** Bio Sample name - multiple Bio Sample names **cannot** be assigned to one SMRT Cell.
 - **Multiple** SMRT Cells can have the **same** Bio Sample name.
 - **All** of the inputs need to already have the appropriate Bio Sample records in their `CollectionMetadata`. If they don't, they are treated as a **single** sample.
2. Create a ReferenceSet from a FASTA file.
 - The ReferenceSet is often already generated and registered in SMRT Link.
 - If the ReferenceSet doesn't exist, use the `dataset create` command to create one:

```
dataset create --type ReferenceSet --name reference_name reference.fasta
```

Launching a multi-sample analysis

```
# Set subreads and ref FASTA
sample1=sample1.subreadset.xml sample2=sample2.subreadset.xml
ref=reference.fasta

pbmm2 align ${ref} ${sample1} sample1.bam --sort --median-filter --sample Sample1
pbmm2 align ${ref} ${sample2} sample2.bam --sort --median-filter --sample Sample2
samtools index sample1.bam
samtools index sample2.bam
pbindex sample1.bam
pbindex sample2.bam
pbsv discover sample1.bam sample1.svsig.gz
pbsv discover sample2.bam sample2.svsig.gz
pbsv call ${ref} sample1.svsig.gz sample2.svsig.gz out.vcf
```

out.vcf: A pbsv VCF output file, where columns starting from column 10 represent structural variants of Sample 1 and Sample 2:

```
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT Sample1 Sample2
chr01 222737 pbsv.INS.1 T TTGGTGTGTTGTTGTTTGTGTTT . PASS
SVTYPE=INS;END=222737;SVLEN=21;SVANN=TANDEM GT:AD:DP 0/1:6,4:10 0/1:6,5:11
```

pbvalidate The `pbvalidate` tool validates that files produced by PacBio software are compliant with PacBio's own internal specifications.

Input files

`pbvalidate` supports the following input formats:

- BAM
- FASTA
- Data Set XML

See [here](#) for further information about each format's requirements.

Usage

```
pbvalidate [-h] [--version] [--log-file LOG_FILE]
           [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug | --quiet | -v]
           [-c] [--quick] [--max MAX_ERRORS]
           [--max-records MAX_RECORDS]
           [--type {BAM,Fasta,AlignmentSet,ConsensusSet,ConsensusAlignmentSet,
SubreadSet,BarcodeSet,ContigSet,ReferenceSet,HdfSubreadSet}]
           [--index] [--strict] [-x XUNIT_OUT] [--unaligned]
           [--unmapped] [--aligned] [--mapped]
           [--contents {SUBREAD,CCS}] [--reference REFERENCE]
           file
```

Required	Description
file	Input BAM, FASTA, or Data Set XML file to validate.

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file LOG_FILE	Writes the log to file. Default (None) will write to stdout.
--log-level	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL.] (Default = CRITICAL)
--debug=False	Alias for setting the log level to DEBUG. (Default = False)
--quiet	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
--verbose, -v	Sets the verbosity level. (Default = None)
--quick	Limits validation to the first 100 records (plus file header); equivalent to --max-records=100. (Default = False)
--max MAX_ERRORS	Exits after MAX_ERRORS were recorded. (Default = None; checks the entire file.)
--max-records MAX_RECORDS	Exits after MAX_RECORDS were inspected. (Default = None; checks the entire file.)
--type	Uses the specified file type instead of guessing. [BAM, Fasta, AlignmentSet, ConsensusSet, ConsensusAlignmentSet, SubreadSet, BarcodeSet, ContigSet, ReferenceSet, HdfSubreadSet] (Default = None)

Options	Description
--index	Requires index files: .fai or .pbi. (Default = False)
--strict	Turns on additional validation, primarily for Data Set XML. (Default = False)

BAM options	Description
--unaligned	Specifies that the file should contain only unmapped alignments. (Default = None, no requirement.)
--unmapped	Alias for --unaligned. (Default = None)
--aligned	Specifies that the file should contain only mapped alignments. (Default = None, no requirement.)
--mapped	Alias for --aligned. (Default = None)
--contents	Enforces the read type: [SUBREAD, CCS] (Default = None)
--reference REFERENCE	Specifies the path to an optional reference FASTA file, used for additional validation of mapped BAM records. (Default = None)

Examples

To validate a BAM file:

```
pbvalidate in.subreads.bam
```

To validate a FASTA file:

```
pbvalidate in.fasta
```

To validate a Data Set XML file:

```
pbvalidate in.subreadset.xml
```

To validate a BAM file and its index file (.pbi):

```
pbvalidate --index in.subreads.bam
```

To validate a BAM file and exit after 10 errors are detected:

```
pbvalidate --max 10 in.subreads.bam
```

To validate up to 100 records in a BAM file:

```
pbvalidate --max-records 100 in.subreads.bam
```

To validate up to 100 records in a BAM file
(equivalent to --max-records=100):

```
pbvalidate --quick in.subreads.bam
```

To validate a BAM file, using a specified log level:

```
pbvalidate --log-level=INFO in.subreads.bam
```

To validate a BAM file and write log messages to a file rather than to stdout:

```
pbvalidate --log-file validation_results.log in.subreads.bam
```


pigeon The `pigeon` tool is a PacBio transcript toolkit that contains tools to classify and filter full-length transcript isoforms into categories (see table below) against a reference annotation.

`pigeon` is based on the [SQANTI3](#) software, and `pigeon` output is compatible with downstream analysis with the [Seurat](#) software.

`pigeon` characterizes isoforms obtained by full-length transcript sequencing. `pigeon` combines the long read-defined transcripts with the reference annotation as well as with other orthogonal data to provide a wide range of descriptors of transcript quality. `pigeon` then generates a comprehensive report to facilitate quality control and filtering of the isoform models.

Pigeon isoform categories

Category	Description
FSM (Full Splice Match)	The reference and query isoform have the same number of exons and each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
ISM (Incomplete Splice Match)	The query isoform has fewer external exons than the reference, but each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
NIC (Novel In Catalog)	The query isoform does not have a FSM or ISM match, but is using a combination of known donor/acceptor sites.
NNC (Novel Not in Catalog)	The query isoform does not have a FSM or ISM match, and has at least one donor or acceptor site that is not annotated.
Antisense	The query isoform does not have overlap with a same-strand reference gene, but is anti-sense to an annotated gene.
Fusion	The query isoform overlaps two or more reference genes.
More junctions	The query isoform overlaps two or more reference genes, however some junctions are shared by multiple genes.
Genic Intron	The query isoform is completely contained within a reference intron.
Genic Genomic	The query isoform overlaps with introns and exons.
Intergenic	The query isoform is in the intergenic region.

Usage

```
pigeon [options] <index,classify,filter,report,make-seurat>
```

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of <code>stdout</code> .
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)

index command: Creates indices for sorted genomic annotation files. (This is **required** for input to the classification stage.) Acceptable annotation files include GTF for annotations and CAGE peak BED files.

Usage

```
pigeon index [options] <input file>
```

Examples

```
pigeon index gencode.v25.annotation.gtf
pigeon index refTSS_v3.1_human_coordinate.hg38.bed
```

Required	Description
Input file	Annotation text file. Annotations must be grouped by reference, with the name in the first column, such as BED or GTF.

Options	Description
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of <code>stdout</code> .
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)

sort command: Sorts the mapped collapsed GFF file. (This is **required** for input to the classification stage.)

Usage

```
pigeon sort [options] <Input file>
```

Example

```
pigeon sort collapsed.gff -o sorted.gf
```

Required	Description
Input file	Mapped collapsed isoform GFF file.

Options	Description
-o	Specifies the output file name. If not specified, the output file name is the input file name with the suffix <code>.sorted</code> .
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of <code>stdout</code> .
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)

classify command: Classifies isoforms according to their structural categories and detects novel isoforms, as compared to known genome annotations.

Usage

```
pigeon classify [options] <isoforms.gff> <annotations.gtf> <reference.fa>
```

Required	Description
isoforms.gff	Sorted collapsed GFF file containing the Isoforms to classify.
annotations.gtf	GTF file containing reference annotations. Note: The associated .pgi file is required . (Use the <code>pigeon index</code> command to generate one.)
reference.fa	FASTA file containing references; this may be gzipped [.gz]. Note: The associated .fai file is required . (Use the <code>samtools faidx</code> command to generate one.)

Input/output options	Description
-r, --ref	Specifies a ReferenceSet dataset XML file. May be used in place of the reference annotation and sequence positional arguments, but the dataset must provide these inputs. May (optionally) provide supplemental reference resources such as the polyA motif and CAGE peak input files.
-d, --out-dir	Specifies the destination directory for all output files. (Default = [.])
-o, --out-prefix	Specifies a prefix for all non-matrix output files. If not specified, the primary input isoform file is used to determine the prefix. Example: An input file of <code>my_isoforms.gff</code> yields the output file name prefix <code>my_isoform</code> .
--fl, --flnc	Specifies the name of a full-length PacBio abundance file.
--cage-peak	Specifies the name of a CAGE (Cap Analysis of Gene Expression) data file in BED format. Note: The associated .pgi file is required . (Use the <code>pigeon index</code> command to generate one.) See here for more information on CAGE.
--poly-a	Specifies the name of a text file containing a ranked list of polyA motifs, one per line.
-c, --coverage	Specifies junction coverage data. Associated .pgi file is required. Use <code>pigeon index</code> to generate one.
--gene-id	Specifies the use of the gene ID instead of the gene name, such as <code>ENSG00000206195.11</code> rather than <code>DUXAP8</code> .

Algorithm setting	Description
--sites	Specifies a comma-separated list of splice sites to be considered as canonical. (Default = <code>GTAG, GCAG, ATAC</code>)
--window	Specifies the size of the window (in base pairs) in the genomic DNA screened for Adenine content downstream of TTS. (Default = 20)
--min-ref-length	Specifies the minimum reference transcript length, in base pairs. (Default = 200)
--is-fusion	Specifies that the inputs are fusion isoforms.
--polya-search-distance	Specifies the distance (in base pairs) to look upstream of the 3' end for the highest ranking polyA motif signal. (Default = 50)

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file</code>	Logs to a file, instead of <code>stdout</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)

`filter` command: Removes isoform classifications that indicate amplification artifacts such as intrapriming or template switching.
Note: `pigeon filter` can **only** be called after `pigeon classify` is first called.

Usage

```
pigeon filter [options] <classification_file>
```

Examples

```
pigeon filter classification.txt
```

```
pigeon filter classification.txt --isoforms sorted.gff
```

Required	Description
classification file	Classification.txt output file from <code>pigeon filter</code> .

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file</code>	Logs to a file, instead of <code>stdout</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)
<code>--isoforms</code>	Specifies the generation of a filtered GFF. This requires that you specify the GFF file that was used as input to <code>pigeon classify</code> .

`report` command: Reports gene saturation. Gene saturation can be determined by subsampling the classification output and determining the number of unique genes at each subsample size.

Usage

```
pigeon report [options] <classification.txt> <subsample.txt>
```

Example

```
pigeon report <classification.filtered_lite_classification.txt> <saturation.txt>
```

Required	Description
classification.txt	Input classification text file. Could be after <code>pigeon classify</code> or <code>pigeon filter</code> .
subsample.txt	Output subsampling text report file.

Options	Description
-s, --sub-sample-increment	Specifies the number of reads between subsampling data points. (Default = 10000)
-h, --help	Displays help information and exits.
--version	Displays program version number and exits.
--log-file	Logs to a file, instead of <code>stdout</code> .
--log-level	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL] (Default = WARN)

`make-seurat` command: Generates the following output files required to run tertiary analysis using the [Seurat](#) software:

- barcodes.tsv
- genes.tsv
- matrix.mtx
- info.csv
- annotated.info.csv

Usage

```
pigeon make-seurat [options] <classification.txt> -a <annotation GTF|XML> --dedup
<dedup fasta> --group <collapse group TXT>
```

Example

```
pigeon make-seurat --dedup dedup.fasta --group
scisoseq.mapped_transcripts.collapse.group.txt
scisoseq_classification.filtered_lite_classification.txt -d output -a
absolutized.referenceset.xml
```

Required	Description
classification.txt	Text file containing transcript classifications. Could be generated by <code>pigeon classify</code> or <code>pigeon filter</code> .
--dedup	Specifies a molecule FASTA file generated by <code>isoseq3 groupdedup</code> or <code>isoseq3 dedup</code> .
-g, --group	Specifies an output groups text file generated by <code>isoseq collapse</code> .
-a, --annotation	Specifies reference annotations for Seurat gene matrix. May be a GTF/ GFF or Reference-Set XML file.

Input/output options	Description
-o, --out-prefix	Specifies a prefix for all non-matrix files. If not specified, the primary input classification file is used to determine the prefix. Example: <code>my_isoforms_classification.txt</code> input yields an output prefix of <code>my_isoforms</code> .

Input/output options	Description
<code>-d, --out-dir</code>	Specifies the destination directory for all output files. (Default = [.])

Matrix options	Description
<code>--keep-novel-genes</code>	Specifies that novel genes be kept in the matrix output.
<code>--keep-ribo-mito-genes</code>	Specifies that ribosomal and mitochondrial genes be kept in the matrix output.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file</code>	Logs to a file, instead of stdout.
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL.] (Default = WARN)

primrose The `primrose` tool analyzes the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.

`primrose` uses a convolution neural network (CNN) to predict the methylation state (5mC) of each CpG in a HiFi read. Methylation is assumed to be symmetric between strands with output reported in the forward direction with respect to the HiFi read sequence. The output uses the `Mm` and `Ml` tags as defined in the SAM Format Optional Fields Specification. (See [here](#) for details.)

Algorithm

The CNN is trained using invitro modified controls of methylated and unmethylated human DNA. The unmethylated control comprises a human shotgun library that has undergone whole genome amplification (WGA), a process that includes PCR, and therefore removes all modifications. The methylated control is generated by subjecting a human WGS library to invitro methylation using a CpG methyltransferase enzyme (*M. SssI*). Kinetic data, pulse width and inter-pulse distance, over a 16 base pair window for both the forward and reverse strand, is used as input to the CNN. The output of the CNN is a probability scale measure of whether the CpG is symmetrically 5mC-modified.

Usage

```
primrose [options] <HiFi INPUT> <HiFi OUTPUT>
```

Required	Description
HiFi INPUT	Input BAM file or ConsensusReadSet XML file.
HiFi OUPUT	Output BAM file or ConsensusReadSet XML file.

Options	Description
<code>--keep-kinetics</code>	Specifies that the kinetics tracks (IPD and PulseWidth records) <code>fi</code> , <code>fp</code> , <code>fn</code> , <code>ri</code> , <code>rp</code> and <code>rn</code> are included in the output BAM file.
<code>--min-passes</code>	Specifies the minimum number of passes. (Default = 3)
<code>--model</code>	Specifies a path to a trained TensorFlow model directory, or to an exported ONNX model file.
<code>-h</code> , <code>--help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-j</code> , <code>--num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stderr</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARNING, , FATAL.] (Default = WARNING)

Example

```
primrose --log-level INFO movie.hifi.bam movie.hifi_5mc.bam
```

runqc-reports The `runqc-reports` tool generates up to five different Run QC reports, depending on Data Set type: Raw Data, Adapters, Loading, Control, and CCS reads. Generating a complete set of reports requires the presence of an `sts.xml` resource in the Data Set, but either the CCS Analysis report (or a fallback Subreads report) will **always** be generated. All report JSON and plot PNG files are written to the current working directory, unless otherwise specified.

Usage

```
runqc-reports [-h] [--version] [--log-file LOG_FILE]
               [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}]
               [| --debug | --quiet | -v]
               [-o OUTPUT_DIR]
               dataset_xml
```

Required	Description
<code>dataset_xml</code>	Input SubreadSet or ConsensusReadSet XML, which must contain an <code>sts.xml</code> resource for the full Run QC report set to be generated.
<code>-o OUTPUT_DIR</code>	Output report directory. (Default = Current working directory)

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stdout</code> .
<code>--log-level</code>	Specifies the log level; values are <code>[DEBUG, INFO, WARNING, ERROR, CRITICAL.]</code> (Default = <code>WARNING</code>)
<code>--debug</code>	Alias for setting the log level to <code>DEBUG</code> . (Default = <code>False</code>)
<code>--quiet</code>	Alias for setting the log level to <code>CRITICAL</code> to suppress output. (Default = <code>False</code>)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = <code>None</code>)

Examples

```
runqc-reports moviename.subreadset.xml
```

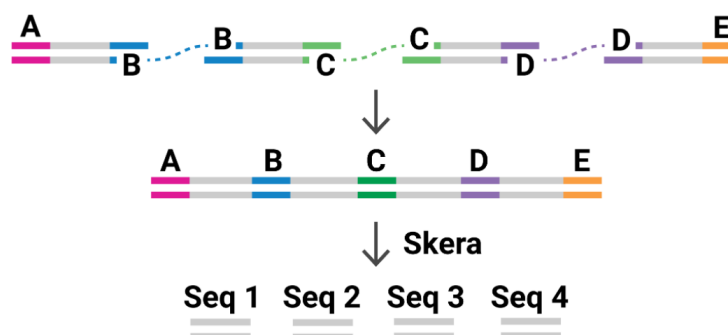
```
runqc-reports moviename.consensusreadset.xml
```


skera The *skera* tool splits arrayed HiFi reads at adapter positions, generating segmented reads (S-reads) which are the individual fragments. For each input HiFi read, the tool creates multiple BAM records, one for each fragment. An arrayed HiFi read can contain many fragments (or S-reads).

The *skera* tool also includes a command to reconstitute the original parent reads from input fragments.

Segmented reads

An arrayed HiFi read is comprised of multiple fragments. The figure below displays an arrayed HiFi read with four fragments (Seq 1 through Seq 4).



Segmented read BAM header

Arrayed HiFi reads have a `READTYPE` of `CCS` in the `DS` (Description) tag in the BAM header. After running *skera split*, segmented reads (S-reads) will have a `READTYPE` of `SEGMENT`. Additionally, a `SOURCE` key is added with a value of `CCS`.

Header DS tag for arrayed HiFi read:

```
DS:READTYPE=CCS;
```

Header DS tag for segmented read (S-Read):

```
DS:READTYPE=SEGMENT; SOURCE=CCS;
```

Segmented read names

For segmented reads (S-reads), the base `QNAME` follows the CCS read conventions, while also appending the 0-based coordinate interval (`qStart`, `qEnd`) that represents its span within the arrayed HiFi read.

Default:

```
{movieName}/{holeNumber}/ccs/{qStart}_{qEnd}
```

If the CCS read is split by strand:

```
{movieName}/{holeNumber}/ccs/fwd/{qStart}_{qEnd}
{movieName}/{holeNumber}/ccs/rev/{qStart}_{qEnd}
```

BAM tag	Type	Description	Example
di	Integer	Index of the read segment.	di:i:0
qs	Integer	qStart of the segment in the parent read.	qs:i:16
qe	Integer	qEnd of the segment in the parent read.	qe:i:450
dl	Integer	Leading segmentation adapter index.	dl:i:0
dr	Integer	Trailing segmentation adapter index.	dr:i:1
ds	Binary	Binary JSON file.	ds:b:10,21,23

Notes:

- All tags of type `Integer` are 0-based.
- `qs` and `qe` tags are left open, right closed.
- `dl` and `dr` tags match the ordering of adapter sequences in the user input Adapter FASTA.
- The `ds` tag contains a number of fields used for supporting the `skera undo` command, which are **not** intended for general use.

`skera split` command: Splits sequences at adapter locations.

Usage

```
skera split [options] <Input dataset> <ADAPTERS.fasta> <Output dataset>
```

Required	Description
Input dataset	Input HiFi reads in BAM/ConsensusReadSet XML format.
ADAPTERS.fasta	Segmentation adapters FASTA or AdapterSet XML file containing segmentation adapters. If not specified, the adapters specified in the XML metadata are used. Segmentation adapters must be ordered in the expected order of adapters in the reads. There should be one entry per adapter (forward or reverse-complement orientation) with no overlapping adapter sequences. Duplicate names or sequences are not allowed. Example: >A AGCTTACTTGTGAAGA >B ACTTGTAAGCTGTCTA >C ACTCTGTCAGGTCCGA >D ACCTCCTCCTCCAGAA >E >AACCGGACACACTTAG
Output dataset	Output S-reads in BAM/ConsensusReadSet XML format.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stderr</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL] . (Default = WARN)

`skera undo` command: Reconstitutes the original parent reads from input fragments (S-reads). **Note:** To run `skera undo`, **both** the passing and `non_passing` BAM files **must** be present in the same directory.

Usage

`skera undo [options] <Input dataset> <Output dataset>`

Required	Description
Input dataset	<code>skera split</code> reads in BAM/ConsensusReadSet XML format.
Output dataset	Arrayed HiFi reads in BAM/ConsensusReadSet XML format.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>-j, --num-threads</code>	Specifies the number of threads to use when processing; 0 means autodetection. (Default = 0)
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to <code>stderr</code> .
<code>--log-level</code>	Specifies the log level; values are [TRACE, DEBUG, INFO, WARN, FATAL] . (Default = WARN)

Input files for `skera split`

- `*.bam`: Arrayed HiFi reads.
- `<adapter>.fasta`: A FASTA file containing segmentation adapters.

Output files for `skera split`

- `*.bam`: BAM file containing S-reads that were segmented from arrayed HiFi reads.
- `*.non_passing.bam`: BAM file containing HiFi reads that failed segmentation. Any arrayed HiFi read in which at least one segment is flanked by adapter sequences that are out of order (as specified by the segmentation adapter FASTA file) is output to the `*non_passing.bam` file to enable the `skera undo` command.
Note: To run `skera undo`, **both** the passing and `non_passing` BAM files **must** be present in the same directory.
- `*.summary.json`, `*summary.csv`: Contains the following summary statistics:

- **Reads:** The number of input arrayed HiFi reads.
- **Segmented Reads (S-reads):** The number of generated S-reads.
- **Mean Length of S-reads:** The mean read length of the generated S-reads.
- **Percentage of Reads with Full Array:** The percentage of input reads containing all adapter sequences in the order listed in the segmentation adapter FASTA file.
- **Mean Array Size:** The mean number of fragments (or S-reads) found in the input reads.
- ***.ligations.csv:** Contains counts of adapter ligations found in arrayed HiFi reads. For each sequence found between two adapters, a ligation between two adapters is counted. The ligations are counted in the order of the adapters in the arrayed HiFi read (`adapter_1`, `adapter_2`).
- ***.read_lengths.csv:** Contains the HiFi read length, S-Read length, and array size for each HiFi read.

Examples

```
skera split <movie>.hifi_reads.bam adapters.fasta <movie>.skera.bam
```

```
skera undo <movie>.skera.bam <movie>.undo.bam
```

summarize Modifications

The `summarizeModifications` tool generates a GFF summary file (`alignment_summary.gff`) from the output of base modification analysis (for example, `ipdSummary`) combined with the coverage summary GFF generated by resequencing pipelines. This is useful for power users running custom workflows.

Usage

```
summarizeModifications [-h] [--version]
                        [--log-file LOG_FILE]
                        [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL} | --debug
                        | --quiet | -v]
                        modifications alignmentSummary gff_out
```

Input files

- `modifications`: Base Modification GFF file.
- `alignmentSummary`: Alignment Summary GFF file.

Output files

- `gff_out`: Coverage summary for regions (bins) spanning the reference with Base Modification results for each region.

Options	Description
<code>-h, --help</code>	Displays help information and exits.
<code>--version</code>	Displays program version number and exits.
<code>--log-file LOG_FILE</code>	Writes the log to file. Default (None) will write to stdout.
<code>--log-level</code>	Specifies the log level; values are [DEBUG, INFO, WARNING, ERROR, CRITICAL] (Default = INFO)
<code>--debug</code>	Alias for setting the log level to DEBUG. (Default = False)
<code>--quiet</code>	Alias for setting the log level to CRITICAL to suppress output. (Default = False)
<code>--verbose, -v</code>	Sets the verbosity level. (Default = None)

Appendix A - Application entry points and output files

Note: To print information about a specific PacBio workflow, including input files and task options, use the `pbcrumwell show-workflow-details` command, which is available for **all** applications.

Examples

```
pbcrumwell show-workflow-details pb_demux_ccs
pbcrumwell show-workflow-details cromwell.workflows.pb_export_ccs
```

(The prefix `cromwell.workflows` is optional.)

CCS Analysis **Analysis application name:** `cromwell.workflows.pb_ccs`

Entry point

```
:id: eid_subread
:name: Entry eid subread
:fileTypeId: PacBio.DataSet.SubreadSet
```

Key output files

File name	Datastore SourceId
FASTQ file	<code>ccs_fastq_out</code>
FASTA file	<code>ccs_fasta_out</code>
<movienam>.hifi.reads.bam file	<code>ccs_bam_out</code>
Consensus Sequences	<code>pb_ccs.ccsxml</code>
CCS Analysis Statistics	<code>pb_ccs.report_ccs</code>
All Reads (BAM)	<code>reads_bam</code>
<movienam>.hifi.reads.fasta	<code>ccs_fasta_out</code>
<movienam>.hifi.reads.fastq	<code>ccs_fastq_out</code>

Demultiplex Barcodes **Analysis application name:** `cromwell.workflows.pb_demux_ccs`

Entry points

```
:id: eid_ccs
:name: Entry eid ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_barcode
:name: Entry eid barcode
:fileTypeId: PacBio.DataSet.BarcodeSet
```

Key output files

File name	Datastore SourceId
Barcode Report Details	<code>pb_demux_ccs.summary_csv</code>
Demultiplexed Datasets	<code>pb_demux_ccs.demuxed_files_datastore</code>
Unbarcoded Reads	<code>pb_demux_ccs.unbarcoded</code>

Export Reads **Analysis application name:** cromwell.workflows.pb_export_ccs**Entry point**

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet
```

Key output files

File name	Datastore SourceId
<moviename>.hifi_reads.fastq.gz	ccs_fastq_out
<moviename>.hifi_reads.fasta.gz	ccs_fasta_out
<moviename>.hifi_reads.bam.gz	ccs_bam_out

Note: If users select a lower cutoff Phred QV value, the string *hifi* is replaced by the QV value in the file names.

Example: <moviename>.q10.fastq.gz.

Genome Assembly

Analysis application name: cromwell.workflows.pb_assembly_hifi

Entry point

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet
```

Key output files

File name	Datastore SourceId
Final Polished Assembly, Primary Contigs	pb_assembly_hifi.final_primary_contigs_fasta
Final Polished Assembly, Haplotigs	pb_assembly_hifi.final_haplotigs_fasta
List of Circular Contigs	pb_assembly_hifi.circular_contigs
Summary Report	pb_assembly_hifi.report_polished_assembly

HiFi Mapping **Analysis application name:** cromwell.workflows.pb_ccs_subreads**Entry points**

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_ref_dataset
:name: Entry eid_ref_dataset
:fileTypeId: PacBio.DataSet.ReferenceSet
```

Key output files

File name	Datastore SourceId
Mapped reads	pb_align_ccs.mapped
Coverage summary	pb_align_ccs.coverage_gff

HiFiViral SARS-CoV-2 Analysis**Analysis application name:** cromwell.workflows.pb_sars_cov2_kit**Entry points**

```

:id: eid_ccs (HiFi reads, demultiplexed as separate BAM files)
:name: Entry_eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_ref_dataset_2 (Reference Genome)
:name: Entry_eid_ref_dataset
:fileTypeId: PacBio.DataSet.ReferenceSet

```

Key output files

File name	Datastore SourceId
All Samples, Probe Counts TSV	pb_sars_cov2_kit.probe_counts_zip
All Samples, Raw Variant Calls VCF	pb_sars_cov2_kit.raw_vcf_zip
All Samples, Variant Call VCF	pb_sars_cov2_kit.vcf_zip
All Samples, Variant Calls CSV	pb_sars_cov2_kit.variants_csv
All Samples, Consensus Sequence FASTA	pb_sars_cov2_kit.fasta_zip
All Samples, Consensus Sequence By Fragments FASTA	pb_sars_cov2_kit.frag_fasta_zip
All Samples, Aligned Reads BAM	pb_sars_cov2_kit.mapped_zip
All Samples, Consensus Sequence Aligned BAM	pb_sars_cov2_kit.aligned_frag_zip
All Samples, Trimmed HiFi reads FASTQ	pb_sars_cov2_kit.trimmed_zip
Failed Sample Info	pb_sars_cov2_kit.sample_failures_csv
Failed Sample Analysis Logs	pb_sars_cov2_kit.errors_zip
Demultiplex Summaries	pb_sars_cov2_kit.lima_summary_zip
Sample Summary Table CSV	pb_sars_cov2_kit.summary_csv
All Samples, Genome Coverage Plots	pb_sars_cov2_kit.coverage_png_zip

Iso-Seq Analysis**Analysis application name:** cromwell.workflows.pb_isoseq3_cconly**Entry points**

```

:id: eid_ccs
:name: Reads
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_barcode
:name: Primers
:fileTypeId: PacBio.DataSet.BarcodeSet

:id: eid_ref_dataset
:name: Reference (Optional)
:fileTypeId: PacBio.DataSet.ReferenceSet

```


Key output files

File name	Datastore SourceId
Collapsed Filtered Isoforms FASTQ	pb_isoseq3_ccsonly.collapse_fastq
Collapsed Filtered Isoforms GFF	pb_isoseq3_ccsonly.collapse_gff
Group TXT	pb_isoseq3_ccsonly.collapse_group
Abundance TXT	pb_isoseq3_ccsonly.collapse_abundance
Read Stat TXT	pb_isoseq3_ccsonly.collapse_readstat
Collapsed Isoforms Abundance TXT (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.collapse_abundance Separate clusters, one per barcoded sample.
Collapsed Isoforms Abundance TXT (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.collapse_abundance Pooled clusters, one per barcoded sample.
High-Quality Transcripts	pb_isoseq3_ccsonly.hq_fastq
Low-Quality Transcripts	pb_isoseq3_ccsonly.lq_fastq
High-Quality Transcripts Counts (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.barcode_overview_report Separate clusters, one per barcoded sample.
High-Quality Transcripts Counts (Files are numbered consecutively, 1 for each barcoded sample.)	pb_isoseq3_ccsonly.barcode_overview_report Pooled clusters, one per barcoded sample.
CCS reads FASTQ	pb_isoseq3_ccsonly.ccs_fastq_zip
Full-length CCS reads	pb_isoseq3._ccsonly.flnc_bam
Polished Report	pb_isoseq3._ccsonly.polish_report_csv
Cluster Report	pb_isoseq3._ccsonly.report_isoseq

**Mark PCR
Duplicates****Analysis application name:** cromwell.workflows.pb_mark_duplicates**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

```

Key output files

File name	Datastore SourceId
PCR Duplicates	pb_mark_duplicates.duplicates
Deduplicated reads	pb_mark_duplicates.deduplicated In the SMRT Link UI, this displays as <ORIGINAL_DATASET_NAME> (deduplicated).

**Microbial
Genome
Analysis****Analysis application name:** cromwell.workflows.pb_microbial_analysis**Entry point**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusreadSet

```

Key output files

File name	Datastore SourceId
Final Polished Assembly	pb_microbial_analysis.assembly_fasta
Final Polished Assembly Index	pb_microbial_analysis.assembly_fasta.fai
Final Polished Assembly for NCBI	pb_microbial_analysis.ncbi_fasta
Mapped BAM	pb_microbial_analysis.mapped
Target sequences used to produce the Mapped BAM	pb_microbial_analysis.mapped_target_fasta
Target sequence Index used to produce the Mapped BAM	pb_microbial_analysis.mapped_target_fasta_fai
List of Circular Contigs	pb_microbial_analysis.circular_list
Coverage Summary	pb_microbial_analysis.coverage_gff
Coverage Report	pb_microbial_analysis.report_coverage
Mapping Statistics Report	pb_microbial_analysis.report_mapping_stats
Mapped BAM Datastore	pb_microbial_analysis.mapped_bam_datastore
Polished Assembly Report	pb_microbial_analysis.report_polished_assembly
Full Kinetics Summary	pb_microbial_analysis.basemods_gff
IPD Ratios	pb_microbial_analysis.basemods_csv
Motifs and Modifications	pb_microbial_analysis.motifs_csv
Motifs and Modifications	pb_microbial_analysis.motifs_gff

**Read
Segmentation****Analysis application name:** cromwell.workflows.pb_segment_reads**Entry points**

```

: id: eid_ccs
: name: HiFi Reads
: fileId: PacBio.DataSet.ConsensusReadSet

: id: Barcode
: name: MAS-Seq Adapters
: fileId: PacBio.DataSet.BarcodeSet

```

Key output files

File name	Datastore SourceId
Input Adapters	pb_segment_reads.adapters_fasta
Report Read Segmentation	pb_segment_reads.report_read_segmentation
Non-Passing Reads	pb_segment_reads.non_passing_bam

File name	Datastore SourceId
Segmented Reads	pb_segment_reads.segmented_reads
Segmented Reads BAM	pb_segment_reads.segmented_reads_bam

Read Segmentation and Single-Cell Iso-Seq Analysis

Analysis application name:

cromwell.workflows.pb_segment_reads_and_sc_isoseq

Entry points

```
:id: eid_ccs
:name: HiFi Reads
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_ref_dataset
:name: Reference Genome
:fileTypeId: PacBio.DataSet.ReferenceSet

:id: eid_barcode
:name: MAS-Seq Adapters
:fileTypeId: PacBio.DataSet.BarcodeSet

:id: eid_barcode_2
:name: Single-Cell Iso-Seq Primers
:fileTypeId: PacBio.DataSet.BarcodeSet
```

Key output files

File name	Datastore SourceId
Input Adapters	pb_segment_reads_and_sc_isoseq.adapters_fasta
Report Read Segmentation	pb_segment_reads_and_sc_isoseq.report_read_segmentation
Barcode statistics	scisoseq.bcstats_report_tsv
Transcript classifications	scisoseq.classification_txt
Transcript classification file, after SQANTI filtering criteria	scisoseq_classification.filtered_lite_classification.txt
Transcript junction file	scisoseq_junctions.txt
Transcript junction file, after SQANTI filtering criteria	scisoseq_classification.filtered_lite_junctions.txt
Unique transcripts, sorted GFF file	scisoseq_transcripts.sorted.gff
Unique transcripts, after SQANTI filtering criteria, GFF file	scisoseq_transcripts.sorted.filtered_lite.gff
Segmented reads, unmapped BAM file	segmented.bam
Reads that did not pass segmentation, unmapped BAM file	segmented.non_passing.bam
Read segmentation report	read_segmentation.report.json
Deduplicated reads mapped BAM file and index file	scisoseq.mapped.bam, scisoseq.mapped.bam.bai
Deduplicated reads, unmapped BAM file	scisoseq.5p--3p.tagged.refined.corrected.sorted.dedup.bam

File name	Datastore SourceId
Single-cell isoform and gene matrix, gzipped	scisoseq.seurat_info.tar.gz

**Single-Cell
Iso-Seq
Analysis**

Analysis application name: cromwell.workflows.pb_sc_isoseq

Entry points

```
:id: eid_ccs
:name: HiFi Reads
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_ref_dataset
:name: Reference Genome
:fileTypeId: PacBio.DataSet.ReferenceSet

:id: eid_barcode
:name: Single-Cell Iso-Seq Primers
:fileTypeId: PacBio.DataSet.BarcodeSet
```

Key output files

File name	Datastore SourceId
Barcode statistics	scisoseq.bcstats_report_tsv
Transcript classifications	scisoseq.classification_txt
Transcript classification file, after SQANTI filtering criteria	scisoseq_classification.filtered_lite_classification.txt
Transcript junction file	scisoseq_junctions.txt
Transcript junction file, after SQANTI filtering criteria	scisoseq_classification.filtered_lite_junctions.txt
Unique transcripts, sorted GFF file	scisoseq_transcripts.sorted.gff
Unique transcripts, after SQANTI filtering criteria, GFF file	scisoseq_transcripts.sorted.filtered_lite.gff
Segmented reads, unmapped BAM file	segmented.bam
Reads that did not pass segmentation, unmapped BAM file	segmented.non_passing.bam
Read segmentation report	read_segmentation.report.json
Deduplicated reads mapped BAM file and index file	scisoseq.mapped.bam, scisoseq.mapped.bam.bai
Deduplicated reads, unmapped BAM file	scisoseq.5p--3p.tagged.refined.corrected.sorted.dedup.bam
Single-cell isoform and gene matrix, gzipped	scisoseq.seurat_info.tar.gz

**Structural
Variant Calling****Analysis application name:** cromwell.workflows.pb_sv_ccs**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_ref_dataset
: name: Entry eid_ref_dataset
: fileId: PacBio.DataSet.ReferenceSet

```

Key output files

File name	Datastore SourceId
Structural Variants	pb_sv_ccs.variants
Aligned reads (Bio Sample Name)	pb_sv_ccs.alignments_by_sample_datastore

**Trim Ultra-Low
Adapters****Analysis application name:** cromwell.workflows.pb_trim_adapters**Entry points**

```

: id: eid_ccs
: name: Entry eid_ccs
: fileId: PacBio.DataSet.ConsensusReadSet

: id: eid_barcode
: name: Entry eid_barcode
: fileId: PacBio.DataSet.BarcodeSet

```

Note: The barcodes need to be a single primer sequence.**Key output files**

File name	Datastore SourceId
Reads Missing Adapters	pb_trim_adapters.unbarcoded
PCR Adapter Data CSV	pb_trim_adapters.summary_csv
Trimmed reads	pb_trim_adapters.trimmed In the SMRT Link UI, this displays as <ORIGINAL_DATASET_NAME> (trimmed).

5mC CpG Detection **Analysis application name:** cromwell.workflows.pb_detect_methyl

Entry points

```
:id: eid_ccs
:name: Entry eid_ccs
:fileTypeId: PacBio.DataSet.ConsensusReadSet

:id: eid_barcode
:name: Entry eid_barcode
:fileTypeId: PacBio.DataSet.BarcodeSet
```

Key output files

File name	Datastore SourceId
HiFi reads with 5mC Calls	pb_detect_methyl.ccsbam

Appendix B - Third party command-line tools

Following is information on the third-party command-line tools included in the `smrtcnds/bin` subdirectory.

- | | |
|-------------------------------|--|
| bamtools | <ul style="list-style-type: none">• A C++ API and toolkit for reading, writing, and manipulating BAM files.• See https://sourceforge.net/projects/bamtools/ for details. |
| cromwell | <ul style="list-style-type: none">• Scientific workflow engine used to power SMRT Link.• See https://cromwell.readthedocs.io/en/stable/ for details. |
| ipython | <ul style="list-style-type: none">• An interactive shell for using the PacBio API.• See https://ipython.org/ for details. |
| purge_dups | <ul style="list-style-type: none">• Removes haplotigs and contig overlaps in a <i>de novo</i> assembly based on read depth.• See https://github.com/dfguan/purge_dups for details. |
| python | <ul style="list-style-type: none">• An object-oriented programming language.• See https://www.python.org/ for details. |
| samtools,
BCFtools | <ul style="list-style-type: none">• A set of programs for interacting with high-throughput sequencing data in SAM/BAM/CRAM/VCF/BCF2 formats.• See http://www.htslib.org/ for details. |

Appendix C - Microbial Genome Analysis advanced options (HiFi reads only)

Use this application to generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb from HiFi (CCS) reads.

The workflow also performs base modification analysis of the assembled genome.

The Microbial Genome Analysis application:

- Includes chromosomal- and plasmid-level *de novo* genome assembly, circularization, polishing, and rotation of the origin of replication for each circular contig.
- Performs base modification detection (m4C and m6A).
- Facilitates assembly of larger genomes (yeast) as well.
- Accepts HiFi read data (BAM format) as input.

The workflow consists of two assembly stages: **chromosomal** and **plasmid**.

Chromosomal stage: Intended for contig assembly of large sequences. This stage uses more stringent filtering (using advanced options) to produce contiguous assemblies of complex regions, but it may miss small sequences in the input sample (such as plasmids.)

Plasmid stage: Intended for a fine-grained assembly. This stage assembles **only** the unmapped and poorly mapped reads. It also relaxes the overlapping parameters, using advanced options.

Both stages use an automated random subsampling process to reduce the input Data Set for assembly (by default to 100x). Note that the subsampling is **only** applied to the contig construction process, while the alignment and reports stages of the workflow still uses the **full** input Data Set.

Available options for the two assembly stages are **identical**. The only differences are:

1. Chromosomal stage parameters are prefixed with:
`ipa2_advanced_options_chrom`. In the SMRT Link interface for the Microbial Genome Analysis application, this corresponds to the **Advanced Assembly Options for chromosomal stage** field.
2. Plasmid stage parameters are prefixed with:
`ipa2_advanced_options_plasmid`. In the SMRT Link interface for the Microbial Genome Analysis application, this corresponds to the **Advanced Assembly Options for plasmid stage** field.

The same sub-options are available to each stage, but the defaults are very different. The current defaults are:

```
ipa2_advanced_options_chrom =
```

```
"config_block_size = 100; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75;"
```

```
ipa2_advanced_options_plasmid =
```

```
"config_block_size = 100; config_ovl_filter_opt = --max-diff 80 --max-cov 100 --min-cov 2 --bestn 10 --min-len 500 --gapFilt --minDepth 4 --idt-stage2 98; config_ovl_min_len = 500; config_seeddb_opt = -k 28 -w 20 --space 0 --use-hpc-seeds-only; config_ovl_opt = --one-hit-per-target --min-idt 98 --min-map-len 500 --min-anchor-span 500 --traceback --mask-hp --mask-repeats --trim --trim-window-size 30 --trim-match-frac 0.75 --smart-hit-per-target --secondary-min-ovl-frac 0.05; config_layout_opt = --allow-circular;"
```

Options are separated by **semicolons**; within each option, parameters are separated by **spaces**.

Users should not need to modify any of default options. If the defaults are modified, workflow behavior could be very different.

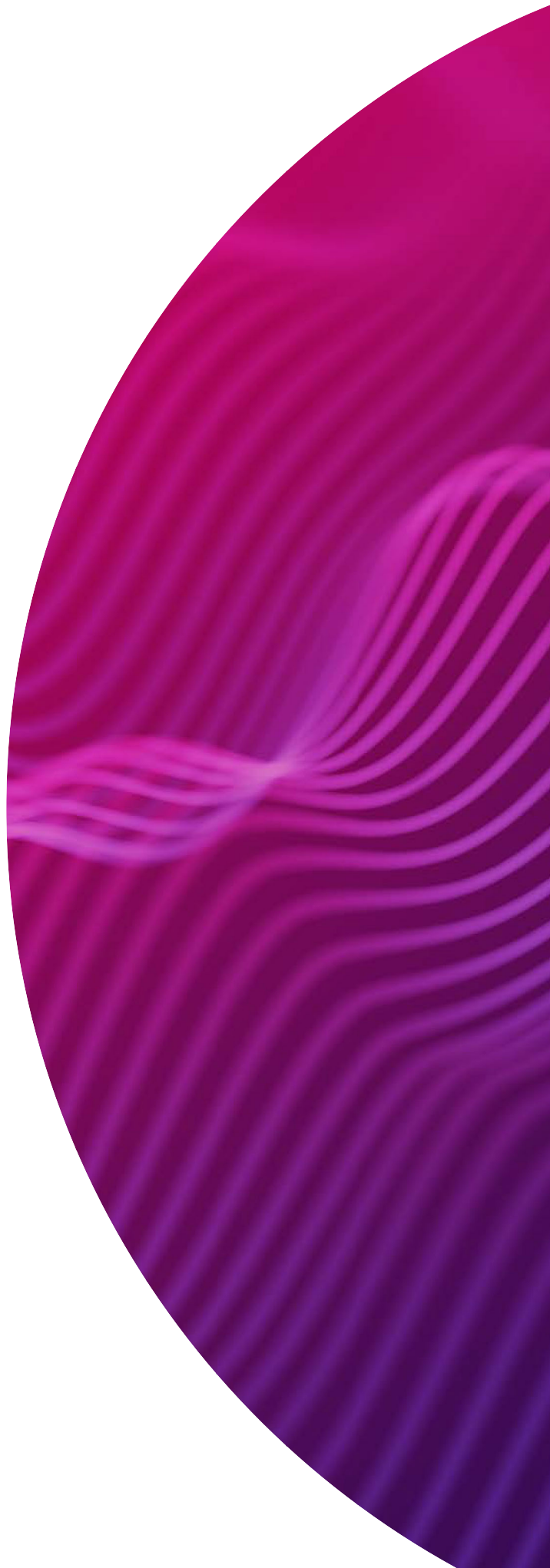
Note: The options available in `ipa2_advanced_options_*` are **exactly** the same as the `config_*` options available for the **Genome Assembly** tool. See [“Genome Assembly parameters input files” on page 44](#) for details.

EXHIBIT M



SMRT[®] Link user guide (v11.1)

Sequel[®] II and IIe
systems



Research use only. Not for use in diagnostic procedures.

P/N 102-413-100 Version 02 (November 2022)

© 2022, PacBio. All rights reserved.

Information in this document is subject to change without notice. PacBio assumes no responsibility for any errors or omissions in this document.

PACBIO DISCLAIMS ALL WARRANTIES WITH RESPECT TO THIS DOCUMENT, EXPRESS, STATUTORY, IMPLIED OR OTHERWISE, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NONINFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL PACBIO BE LIABLE, WHETHER IN CONTRACT, TORT, WARRANTY, PURSUANT TO ANY STATUTE, OR ON ANY OTHER BASIS FOR SPECIAL, CONSEQUENTIAL, INCIDENTAL, EXEMPLARY OR INDIRECT DAMAGES IN CONNECTION WITH (OR ARISING FROM) THIS DOCUMENT, WHETHER OR NOT FORESEEABLE AND WHETHER OR NOT PACBIO IS ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Certain notices, terms, conditions and/or use restrictions may pertain to your use of PacBio products and/or third party products. Refer to the applicable PacBio terms and conditions of sale and to the applicable license terms at <http://www.pacificbiosciences.com/licenses.html>.

Trademarks:

Pacific Biosciences, the PacBio logo, PacBio, Circulomics, Omnione, SMRT, SMRTbell, Iso-Seq, Sequel, Nanobind, and SBB are trademarks of Pacific Biosciences of California Inc. (PacBio). All other trademarks are the sole property of their respective owners.

See <https://github.com/broadinstitute/cromwell/blob/develop/LICENSE.txt> for Cromwell redistribution information.

PacBio
1305 O'Brien Drive
Menlo Park, CA 94025
www.pacb.com



SMRT® Link user guide (v11.1)

Introduction	4
Contact information	5
Module menu commands	6
Gear menu commands	6
Sending information to Technical Support	7
Sample Setup	8
Application-based calculations	8
Custom calculations	10
Classic mode	10
Editing or printing calculations	12
Deleting calculations	12
Importing/exporting calculations	12
Run Design	16
Creating a new Run Design	16
Custom Run Designs	19
Advanced options	20
Editing or deleting Run Designs	20
Creating a Run Design by importing a CSV file	21
Run QC	27
Table fields	29
Run settings and metrics	31
Data Management	34
What is a Data Set?	34
Creating a Data Set	35
Viewing Data Set information	36
Copying a Data Set	36
Deleting a Data Set	37
Starting a job from a Data Set	37
Data Set QC reports	37
What is a Project?	39
Data Sets and Projects	39
Creating a Project	39
Editing a Project	40
Deleting a Project	40
Viewing/deleting sequence, reference and barcode data	41
Importing sequence, reference and barcode data	41
Exporting sequence, reference and barcode data	42
SMRT® Analysis	44
Creating and starting a job	44
Starting a job after viewing sequence data	49
Canceling a running job	50

Restarting a failed job	50
Viewing job results	51
Copying and running an existing job	52
Exporting a job	52
Importing a job	52
Summarizing Microbial Genome Analysis jobs	53
PacBio® secondary analysis applications	54
Genome Assembly	56
HiFi Mapping	59
HiFiViral SARS-CoV-2 Analysis	63
Iso-Seq® Analysis	68
Microbial Genome Analysis	75
Read Segmentation and Single-Cell Iso-Seq® Analysis	81
Single-Cell Iso-Seq® Analysis	88
Structural Variant Calling	95
PacBio® data utilities	99
5mC CpG Detection	100
Demultiplex Barcodes	101
Export Reads	107
Mark PCR Duplicates	109
Read Segmentation	111
Trim Ultra-Low Adapters	113
Circular Consensus Sequencing (CCS)	115
Working with barcoded data	118
Step 1: Specify barcode setup & sample names in a Run Design	119
Step 2: Perform the sequencing run	120
Step 3: (Optional) Run the Demultiplex Barcodes data utility	121
Step 4: Run applications using the demultiplexed data as input	122
Demultiplex Barcodes details	123
Automated analysis	126
Creating an Auto Analysis job from SMRT Analysis	126
Creating Auto Analysis from a Run Design	127
HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design	127
Getting information about analyses created by Auto Analysis	128
Visualizing data using IGV	129
Using the PacBio® self-signed SSL certificate	131
Sequel® II system and Sequel Ii system output files	132
Sequel Ii system output files	132
Sequel II system output files	135
Secondary analysis output files	137
Configuration and user management	140
LDAP	140
SSL	140

Adding and deleting SMRT Link users 141

Hardware/software requirements 143

Appendix A - PacBio terminology 144

Appendix B - Data search. 147

Appendix C - BED file format for Target Regions report 149

Appendix D - Additional information in the CCS Data Set Export report. 150

Introduction

This document describes how to use PacBio's SMRT Link software. SMRT Link is the web-based end-to-end workflow manager for Sequel II systems and Sequel IIe systems. SMRT Link includes the following modules:

- **Sample Setup:** Calculate binding and annealing reactions for preparing DNA libraries for use on **all** Sequel II systems and Sequel IIe systems. (See ["Sample Setup" on page 8](#) for details.)
- **Run Design:** Design sequencing runs and create and/or import sample sheets. (See ["Run Design" on page 16](#) for details.)
- **Run QC:** Monitor run progress, status and quality metrics. (See ["Run QC" on page 27](#) for details.)
- **Data Management:** Create Projects and Data Sets; generate QC reports for Data Sets; view, import, or delete sequence, reference, and barcode files. (See ["Data Management" on page 34](#) for details.)
- **SMRT Analysis:** Perform secondary analysis on the basecalled data (such as sequence alignment, variant detection, *de novo* assembly, structural variant calling, and RNA analysis) after a run has completed. (See ["SMRT® Analysis" on page 44](#) for details.)

Note: SMRT Link v11.1 is for use with **Sequel II** systems and **Sequel IIe** systems **only**. If you are using a Sequel system, use an **earlier** version of SMRT Link.

This document also describes:

- The data files generated by the Sequel II system and Sequel IIe systems for each cell transferred to network storage. (See ["Sequel® II system and Sequel IIe system output files" on page 132](#) for details.)
- The data files generated by secondary analysis. (See ["Secondary analysis output files" on page 137](#) for details.)
- Configuration and user management. (See ["Configuration and user management" on page 140](#) for details.)
- SMRT Link client hardware/software requirements. (See ["Hardware/software requirements" on page 143](#) for details.)

Installation of SMRT Link **server** software is discussed in the document **SMRT Link software installation guide (v11.1)**.

New features, fixed issues and known issues are listed in the document **SMRT Link release notes (v11.1)**.

When you first start SMRT Link, you must specify which system you are using: **Sequel II**, or **Sequel IIe**. This choice affects some of the initial values used in the Sample Setup and Run Design modules. In those modules, you can switch between the two Sequel systems as needed.

Users with administrator access can configure SMRT Link to support **all** instrument types.

Contact information

For additional technical support, contact PacBio at support@pacb.com or 1-877-920-PACB (7222).

Using SMRT® Link

You access SMRT Link using the Chrome web browser.

- SMRT Link is **not** available on the instrument – it must be accessed from a remote workstation.
- Depending on how SMRT Link was installed at your site, logging in with a user name and password may be required.
- SMRT Link needs a Secure Sockets Layer (SSL) certificate to ensure a secure connection between the SMRT Link server and your browser using the HTTPS protocol.

If an SSL certificate is **not** installed with SMRT Link, the application will use the PacBio self-signed SSL certificate and will use the HTTP protocol. In this case, **each** user will need to accept the browser security warnings described in [“Using the PacBio® self-signed SSL certificate” on page 131](#).

After accessing SMRT Link, the **home page** displays.



- Click the **PacBio logo** at the top left to navigate back to the SMRT Link home page from within the application.
- Click the **Gear menu** to sign out, configure for the Sequel II system or Sequel IIe system, view version information, or perform administrative functions (Admins **only**).
- Click a module name to access that module. **Sample Setup**, **Run Design**, **Data Management** and **SMRT Analysis** include links to create new Calculations, Run Designs, Data Sets, and jobs. (A **Module** menu displays next to the PacBio logo, allowing you to move between modules.)

- Click **?** to view the SMRT Link online help.
- Select **Sign Out** from the Gear menu to log out of SMRT Link.

Module menu commands

- **Sample Setup:** Displays the Sample Setup module.
- **Run Design:** Displays the Run Design module.
- **Run QC:** Displays the Run QC module.
- **Data Management:** Displays the Data Management module.
- **SMRT Analysis:** Displays the SMRT Analysis module.

Gear menu commands

- **Show Alarms**
 - Displays SMRT Link system-level alarms. To clear alarms, select and click **Clear Alarm** or **Clear All Alarms**.
- **Configure**
 - To specify instruments (Sequel II system, Sequel IIe system) that SMRT Link will be used with, click **Instruments** and check the appropriate box(es).
 - **Admin users only:** Add/delete SMRT Link users and specify their roles. See [“Adding and deleting SMRT Link users” on page 141](#) for details.
 - To specify how numbers are formatted, click **Number Formatting** and select **Period** or **Comma** as the decimal separator.
 - **(Sequel IIe system only)** To specify whether CCS analysis output includes kinetics information (used for epigenetics analysis), click **CCS Analysis Output** and select **Yes** or **No**. This is the default setting for **all** CCS analysis output, unless overwritten in individual Run Designs. **Note:** Adding kinetics information can increase the amount of storage used by the output BAM files by up to **5 times**.
- **About SMRT Link**
 - Displays software version information and available space on the server SMRT Link is connected to.
 - Click **Send** to send configuration information and/or analysis usage information to PacBio Technical Support for help in troubleshooting failed jobs.
 - **Admin users only:** 1) Update the SMRT Link **Chemistry Bundle**, which includes kit and DNA Control Complex names used in the Sample Setup and Run Design modules. 2) Update the SMRT Link **UI Bundle**, which includes changes and bug fixes to the SMRT Link Graphical User Interface or UI for a SMRT Link module.
- **Sign Out**
 - Logs you out and displays the initial login page.

Working with tables

- To sort table columns: Click a column title.
- To see additional columns: Click the > symbol next to a column title.

- To search within a table: Enter a unique search string into the Search field. (For details, see “Appendix B - Data search” on page 147.)

Jobs

Click a column name to sort on

Enter a unique search term

Displaying rows 1 to 12 out of 65

Name	Member Jobs	State	ID	Date Created	Date Updated	Created By	Workflow
detect_methyl_tiny_rhino		SUCCESSFUL	176	2022-02-24, 09:28:00 AM	2022-02-24, 09:33:27 AM	smrtlinktest	5mC CpG Detection
basemods_tiny_hifi_kiwi_hpyl		SUCCESSFUL	183	2022-02-24, 09:28:11 AM	2022-02-24, 09:35:53 AM	smrtlinktest	Base Modification Analysis
basemods_tiny_ecoli_kiwi		SUCCESSFUL	196	2022-02-24, 09:28:27 AM	2022-02-24, 09:38:54 AM	smrtlinktest	Base Modification Analysis (Subre...

Sending information to Technical Support

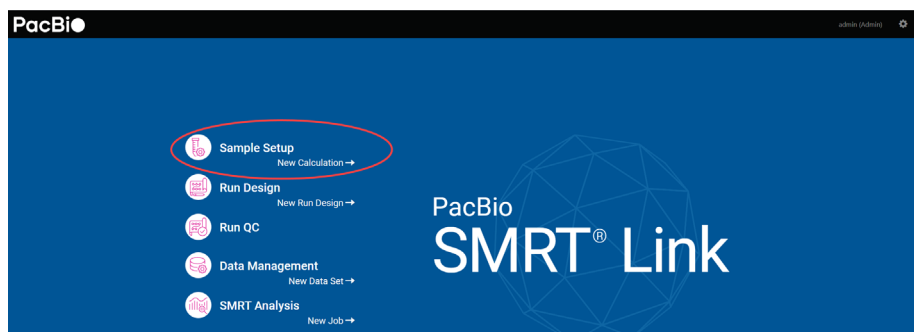
To open a case with PacBio Technical Support, send an email to support@pacb.com.

Troubleshooting information can be sent to PacBio Technical Support two ways:

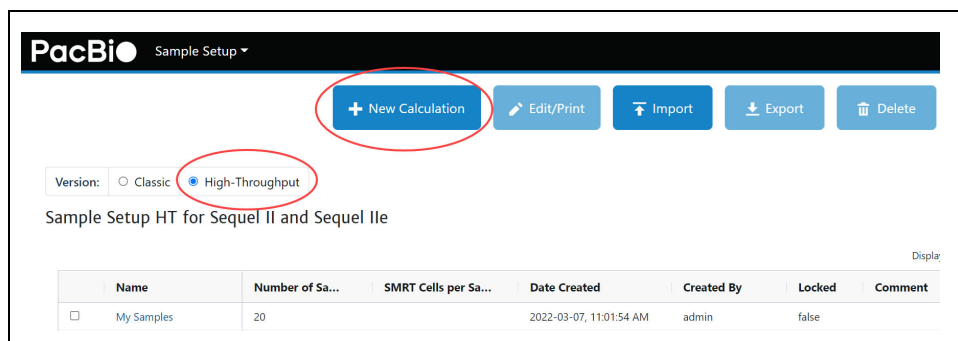
- From the SMRT Link menu: **About > Troubleshooting Information > Send.**
- From a SMRT Link “Failed” analysis Results page: Click **Send Log Files.**

Sample Setup

To prepare your samples for sequencing, use SMRT Link's **Sample Setup** module to generate a customized protocol for primer annealing and polymerase binding to SMRTbell® templates, with subsequent sample cleanup. You can then print the instructions for use in the lab.



1. Access SMRT Link using the Chrome web browser.
2. Select **Sample Setup**.
3. Select **High-Throughput** mode, which provides a more streamlined workflow to efficiently process multiple samples with similar library properties (such as mean insert size and DNA concentration) in parallel. You can also export the calculated values to a CSV file for laboratory automation.
Note: **Classic** mode is provided for legacy support purposes **only**, and is described later in this document.
4. Click **+ New Calculation**.



5. Enter the sample **name**.

Application-based calculations

6. Select a sequencing **application** for the sample. The following fields are **auto-populated** and display in green:
 - Binding Kit
 - Cleanup Anticipated Yield

Whole Genome Sequencing
HiFi Reads
Microbial Assembly
RNA Sequencing
Iso-Seq Method
MAS-Seq single cell
Viral Sequencing
HiFiViral SARS-CoV-2
Adeno-Associated Virus
Metagenomics
Full-Length 16S rRNA Sequencing
Shotgun Metagenomic Profiling or Assembly
Amplicon Sequencing
<3kb Amplicons
>=3kb Amplicons
Other
Custom

7. Enter the **number of samples** for this calculation. Samples should be substantially equivalent to each other; all should have insert sizes and concentrations within +/- 15% of the specified values.
8. Enter the **number of SMRT® Cells** to bind per sample.
9. Enter the available **volume per sample**, in µL. When preparing multiple samples, this should be the **minimum** volume available for any sample.
10. Specify an **insert size**, in base pairs. The insert size is the length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters. This matches the mean insert size for the sample; the size range boundaries are described in the library preparation protocol. Enter the mean insert size of the sample(s).
11. Enter the sample **concentration(s)**, in ng/ul. Note that the acceptable range of input concentrations depends on insert size:

Insert size	Concentration range
10kb and up	20 - 60 ng/ul
3kb - 9999 bp	6 - 20 ng/ul
1.5kb - 2999 bp	3 - 10 ng/ul
500bp - 1499 bp	1 - 3 ng/ul

12. If necessary, edit the **Cleanup anticipated yield**. Adjust this percentage based on previous experience. (Cleanup removes excess primers/polymerase from bound complexes, which results in higher quality data.)
13. Specify the on-plate loading concentration (OPLC), in pM.
14. Specify the **Minimum Pipetting Volume**, in uL. This allows you to set a lower limit on pipetting volumes to use in certain protocol steps, such as sample annealing and binding. We recommend setting this to 1 uL, though in some cases, for example if sample availability is very limited, it may be appropriate to set a value below 1 uL. Some protocol steps include fixed values of 1 uL that will **not** be affected by this setting.
15. Optionally, do one of the following:
 - Click **Copy** to start a **new** sample group using the information entered. Then, edit specific fields for each sample group.
 - Click **Automate** to generate a CSV file. This exports the calculated values to a CSV file for lab automation.

16. To **print** the calculation(s) and instructions, use the browser's Print command (**Ctrl-P**).

Custom calculations

1. To accommodate new or unique sample types, choose **Application > Custom** and enter all settings manually.
2. Click **Set Custom Preset Values** to save any custom application settings you may have specified. The next time you select **Application > Custom**, those settings are retrieved.

Classic mode

Note: Classic mode is provided for legacy support purposes **only**. We **highly recommend** using High-Throughput mode even for single samples.

1. Select **Classic** mode.
2. Select **Sequel II** or **Sequel IIe**.
3. Click **+ New Calculation**.
4. Enter the sample **name**.
5. Select a sequencing **application** for the sample.
6. Enter the available sample **volume**, in μL .
7. Enter the sample **concentration**, in $\text{ng}/\mu\text{L}$.
8. Specify an **insert size**, in base pairs.
9. Select the **Internal Control** version to use for this run from the list, or type in a part number. PacBio **highly** recommends using the Internal Control to help distinguish between sample quality and instrument issues in the event of suboptimal sequencing performance. (**Note:** PacBio **requires** the use of the Internal Control for consumables to be eligible for reimbursement consideration.)
10. If necessary, edit the **Cleanup anticipated yield**. Adjust this percentage based on previous experience. (Cleanup removes excess primers/polymerase from bound complexes, which results in higher quality data.)
11. Specify the on-plate loading concentration (OPLC), in pM.
12. Enter the number of SMRT[®] Cells to bind, at the specified on-plate loading concentration.
13. Rather than leave a small amount of library behind, use the entire library volume available if desired by selecting **Prepare Entire Sample > Yes**. This generates annealing, binding and cleanup instructions for the **entire available sample volume**. The instructions for loading the sample plate will still follow the scale indicated by the specified number of SMRT Cells to run.
14. In the complex cleanup step, enter the pre- and post-cleanup sample DNA quantitation and volume measurement results.

	Sample 1	✓
Volume of Purified Complex (uL)	<input type="text"/>	
Purified Complex Concentration (ng/uL)	<input type="text"/>	
Molar Concentration of Purified Complex (pM)	???	
Ampure Cleanup Yield (%)	???	

15. Optionally, specify an alternative number of cells or on-plate loading concentration (OPLC) for the final sample dilution step. Use this feature, for example, to initially set up a single-SMRT Cell run to test a specific loading concentration prior to conducting a multi-SMRT Cell sequencing run, or to set up a loading titration experiment to optimize the OPLC for your particular sample.

Final Loading Dilution

Reagent	Sample 1	✓	
Warning	Values measured in cleanup step must be entered		
Sequel® Complex Dilution Buffer	0.0 uL		
Prepared sample	0.0 uL		
Diluted Internal Control (Dilution 2)	0.0 uL		
DTT	0.0 uL		
Sequel Additive	0.0 uL		
Total Volume	0.0 uL		
# of SMRTCells requested	5		
Show values for a different number of cells	<input type="text"/>		
Show values for a different OPLC	<input type="text"/>		

Load 115 uL of sample per well and store at 4C for up to 24 hours before use.

16. Optionally, do one of the following:
- Click **Copy** to start a **new** sample using the information entered. Then, edit specific fields for each sample.
 - Click **Remove** to delete the current calculation.
 - Click **Lock** to lock the calculation. This is **required** before samples can be imported into the Run Design module, and also sends a finalized version of the instructions to the server for use in Data Set reports. After locking, no further changes can be made to a calculation. (Click **View** to see the locked instructions.) Locking ensures that calculations are always synchronized with their run time state if a report is generated at a later date. (**Lock** is **only** available if there are one or more samples visible **and** most fields have values entered.)
 - Click the **New Sample** button at the top of the screen to start a new, empty sample.
17. Specify whether to display the **full** instructions, or only the **loading** instructions.

Advanced options

- Specify the **Minimum Pipetting Volume**, in uL. This allows you to set a lower limit on pipetting volumes to use in certain protocol steps, such as sample annealing and binding. We recommend setting this to 1 uL, though in some cases, for example if sample availability is very limited, it may be appropriate to set a value below 1 uL. Some protocol steps include fixed values of 1 uL that will **not** be affected by this setting.
- Specify the **% of Annealing Reaction to Use in Binding**. This accommodates pipetting underage: Due to pipetting issues, volumes may not add up to what they should; a value below 100% helps ensure there will be enough annealed sample for binding.

Editing or printing calculations

1. On the **Sample Setup** screen, select one or more calculation names.
2. Click **Edit/Print**. (**Note:** If the samples use different versions of chemistry, a warning message displays.)
3. Edit the sample(s) as necessary.
4. Specify whether to display the **full** instructions, or only the **loading** instructions.
5. To print the calculation(s), use the browser's **Print** command (Ctrl-P).

Deleting calculations

1. On the **Sample Setup** screen, select one or more calculation names to delete.
2. Click **Delete**.

Importing/exporting calculations

Sample Setup supports importing and exporting calculations in CSV format.

To **import** a new calculation, first find (or create) a calculation **similar** to that you wish to import, then export it in CSV format. You can then customize the exported CSV file as needed, then **import** the modified CSV file.

Note: The content of the CSV file generated using the **Export** button in the Sample Setup home screen is **different** from the content of the CSV file generated using the High-Throughput mode's **Automate** button used for lab automation.

1. Access SMRT Link using the Chrome web browser.
2. Select **Sample Setup**.
3. Select **High-Throughput**.
4. Select an existing calculation.
5. Click **Export**, then click **Download**.
6. Edit the exported calculation in Excel (changing sample names, concentrations, and so on), then save it under a new name.

7. In Sample Setup, click **Import**.
8. Click **Browse**, then select the CSV file you previously modified in Step 6 and click **Open**. If everything is correct, click **Continue**. The imported calculation displays.

Note:

- You can select **multiple** calculations to export to the same CSV file.
- You can also **import** multiple calculations by adding rows to the CSV file.

Following are the fields contained in the CSV-format Calculations file.

Field name	Required	Description
Sample Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only .
System Name	Yes	Must be Sequel II, or Sequel IIE.
Application	Yes	Enter one of the following values: <ul style="list-style-type: none"> • HiFi Reads • Microbial Assembly • Iso-Seq Method • MAS-Seq single cell • HiFiViral SARS-CoV-2 • Adeno-Associated Virus • Full-Length 16S rRNA Sequencing • Shotgun Metagenomic Profiling or Assembly • <3kb Amplicons • >=3kb Amplicons • Custom
Available Starting Sample Volume (uL)	Yes	Enter a positive integer. Units are in microliters.
Starting Sample Concentration (ng/uL)	Yes	Enter a positive integer. Units are in nanograms per microliter.
Insert Size (bp)	Yes	Enter a positive integer. Units are in base pairs.
Control Kit	No	Must be blank or Lxxxxxx101717600123199.
Cleanup Anticipated Yield (%)	No	Enter a positive integer. Note: If Application is set to Custom , this field is required .
On Plate Loading Concentration (pM)	Yes	Enter a positive integer. Units are in parts per million.
Cells to Bind (cells)	Yes	Enter a positive integer.
Prepare Entire Sample	Yes	Enter a Boolean value: true, t, yes, y, false, f, no, or n . Boolean values are not case-sensitive.
Sequencing Primer	Yes	Enter one of the following values: <ul style="list-style-type: none"> • Sequencing Primer v2 • Sequencing Primer v4 • Sequencing Primer v5

Field name	Required	Description
Binding Kit	Yes	For Sequel II/Ile Binding Kits 2.0, 2.1, 2.2, 3.1 and 3.2: <ul style="list-style-type: none"> • Lxxxxxx101780500123199 (2.0) • Lxxxxxx101820500123199 (2.1) • Lxxxxxx101894200123199 (2.2) • Lxxxxxx102194200123199 (3.1) • Lxxxxxx102194100123199 (3.2)
Target Annealing Concentration (nM)	No	Enter a positive integer. Units are in nanomolar. Note: If Application is set to <i>Custom</i> , this field is required .
Target Binding Concentration (nM)	No	Enter a positive integer. Units are in nanomolar. Note: If Application is set to <i>Custom</i> , this field is required .
Target Polymerase Concentration (X)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Binding Time (hours)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Cleanup Bead Type	No	Must be AMPure or ProNex. Note: If Application is set to <i>Custom</i> , this field is required .
Cleanup Bead Concentration (X)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
Minimum Pipetting Volume (uL)	No	Enter a positive integer. Units are in microliters.
Percent of Annealing Reaction To Use In Binding (%)	No	Enter a positive integer. Note: If Application is set to <i>Custom</i> , this field is required .
AMPure Diluted Bound Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
AMPure Diluted Bound Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
AMPure Purified Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
AMPure Purified Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
ProNex Diluted Bound Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
ProNex Diluted Bound Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
ProNex Purified Complex Volume (uL)	No	Enter a positive integer. Units are in microliters.
ProNex Purified Complex Concentration (ng/uL)	No	Enter a positive integer. Units are in nanograms per microliter.
Requested Cells Alternate (cells)	No	Enter a positive integer.
Requested OPLC Alternate (pM)	No	Enter a positive integer. Units are in parts per million.

CSV file general requirements

- Each line in the CSV file represents **one** sample.
- The CSV file may **only** contain ASCII characters. Specifically, it must satisfy the regular expression `/^[\x00-\x7F] *$/g`

Following are the fields contained in the CSV file generated by the **Automate** button in High-Throughput mode. This includes **all** the fields

that display in the Sample Setup page, with the volumes listed in each table easily accessible for liquid handling automation purposes.

Row	Field name
1	Export Version Version number of the file format specification. Allows for scripts to check version numbers to ensure compatibility through subsequent software releases.
2	Instructions Version Version number of SMRT Link, chemistry bundle, and parameters.
3	Sample Group Name
4	Annealing Number of Samples
5	Annealing Sample Volume
6	Annealing Master Mix Volume
7	Annealing Incubation Temperature (C)
8	Annealing Incubation Time (minutes)
9	Polymerase Stock Volume
10	Sequel II Polymerase Dilution Buffer Volume
11	Binding Number of Samples
12	Binding Annealed Sample Volume
13	Binding Master Mix Volume
14	Binding Diluted Polymerase Volume
15	Binding Incubation Temperature (C)
16	Binding Incubation Time (minutes)
17	ICD1 Sequel Complex Dilution Buffer Volume
18	ICD1 Internal Control Stock Volume
19	ICD2 Sequel Complex Dilution Buffer Volume
20	ICD2 Diluted Internal Control (ICD1) Volume
21	ICD3 Sequel Complex Dilution Buffer Volume
22	ICD3 Diluted Internal Control (ICD2) Volume
23	Cleanup S2 Sample Input Volume
24	Cleanup S2 Diluent Volume
25	Cleanup S2 Binding Buffer
26	Cleanup S3 Bead Solution Volume
27	Cleanup S5 Elution Volume
28	Cleanup S5 Elution Buffer
28	Final Loading Number of Samples
30	Final Loading Prepared Sample Volume
31	Final Loading Diluted Internal Control (ICD3) Volume
32	Final Loading Volume (micro-liter)

Run Design

Use SMRT Link's **Run Design** module to create, edit, or import Run Designs. A **Run Design** specifies:

- The samples, reagents, and SMRT Cells to include in the sequencing run.
- The run parameters such as movie time and loading to use for the sample.

The Run Design then becomes available from the **Sequel Instrument Control Software (ICS)**, which is the instrument touchscreen software used to select a Run Design, load the instrument, and then start the run.

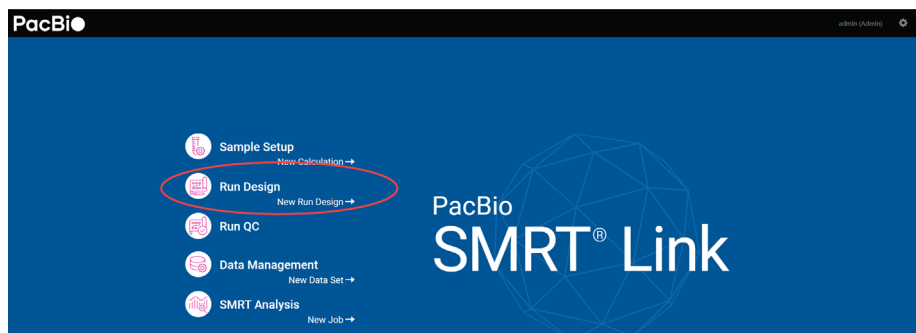
Run Designs created in SMRT Link are accessible from **all** Sequel II systems and Sequel IIe systems linked to the same SMRT Link server.

SMRT Link includes two different ways to create a Run Design:

- Use SMRT Link's **Run Design** module to create a new Run Design.
- Create a CSV file, then import it using SMRT Link's **Run Design** module.

Note: To create a run design, **either** use the Run Design screen, **or** import a CSV file. Do **not** mix the two methods.

Creating a new Run Design



1. Access SMRT Link using the Chrome web browser.
2. Select **Run Design**.
3. Run Designs can be sorted and searched for:
 - To sort Run Designs, click a **column title**.
 - To search for a Run Design, enter a unique search string into the **Search** field.
4. To initiate a new Run Design, click **+ Create New Design**.

5. Specify if this Run Design is to be used with a Sequel II system or a Sequel IIe system. This affects the initial default values.
6. Enter a **Run Name**. (The software creates a new run name based on the current date and time; edit the name as needed.)
7. **(Optional)** Enter **Run Comments**, **Experiment Name**, and **Experiment ID** as needed. (**Note:** Experiment ID **must** be alphanumeric.)
8. **(Optional)** Click **Select Sample** to import information from a previously-created Sample Setup entry. The following fields are auto-populated as appropriate:
 - Sample Name
 - SMRTbell Adapter Design
 - Binding Kit
 - DNA Control Complex
 - Insert Size
 - On-Plate Loading Concentration

Application-based Run Designs

9. Select a sequencing **application** from the list. The following fields are auto-populated, and display in green:
 - Template Prep Kit
 - Binding Kit
 - Sequencing Kit
 - DNA Control Complex
 - Movie Time per SMRT Cell (hours)
 - Pre-Extension Time (hours)
10. Enter a **Well Sample Name**. (This is the name of the sequencing library loaded into one well. **Example:** HG002_2019_11_02_10K)
11. Enter a **Bio Sample Name**. (This is the name of the biological sample contained in the sequencing library, such as HG002. See [“Working with barcoded data” on page 118](#) for details.)
12. (Optional) Enter **Sample Comments**.
13. Specify the **well position** used for this sample: Click the icon to the right of the entry field and choose a plate position.
14. Specify an **insert size** (500 base pairs minimum). The insert size is the length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters. This matches the average insert size for the sample; the size range boundaries are described in the library preparation protocol. **Note:** The default insert size for Subreads is 30,000; 10,000 for CCS reads.
15. Specify the **On-Plate loading concentration** (OPLC), in picomolarity.
16. (Optional) If you are using **barcoded samples**, see [“Step 1: Specify the barcode setup and sample names in a Run Design” on page 119](#) for instructions. For details on secondary analysis of barcoded samples, see [“Demultiplex Barcodes” on page 101](#).
17. Sample options:
 - Click **Copy**. This starts a new sample, using the values entered in the first sample.
 - Click **Delete**. This deletes the current sample.
 - Click **Add Sample**. This starts a new, empty sample.
18. After filling in all the samples, click **Save** - this saves the entire Run Design. The new Run Design displays on the main Run Design page.
19. Click **View Summary** to view a table summarizing the entire Run Design. The Run Design file is now imported and available for selection in Sequel ICS on the instrument.
20. (Optional) **Auto Analysis** allows a specific analysis to be **automatically** run after a sequencing run has finished and the data transferred to the SMRT Link server. See [“Automated analysis” on page 126](#) for details.

If **MAS-Seq single cell** is selected as the application, the following additional **Auto Analysis** fields display:

- **Launch Read Segmentation:** Specify whether or not to split arrayed HiFi reads at MAS adapter positions, generating segmented reads (S-reads). If you specify **No**, you **cannot** launch Single-Cell Iso-Seq[®] Analysis.
- **Adapter Options:** Specify the MAS segmentation adapter choice to be **MAS16** (default) or **MAS12**.
- **Launch Single-Cell Iso-Seq Analysis:** Specify whether or not to enable analysis and functional characterization of full-length transcript isoforms with additional single-cell information, including single-cell barcodes and unique molecular identifiers (UMIs). To run this analysis, the **Read Segmentation** utility **must** be run first.
- **Sample Type:** Specify whether to use the **Human** or **Mouse** default reference genome and annotation set to align high-quality isoforms to, and to collapse isoforms mapped to the same genomic loci.
- **Analysis Name:** Specify the name of the analysis to be generated.

Custom Run Designs

To accommodate new or unique Run Designs, choose **Application > Custom** and enter all parameters manually. (See [here](#) for recommendations based on the analysis application used.)

- **SMRTbell Adapter Design, Template Prep Kit, Binding Kit, or Sequencing Kit:** Select one from the list, or type in a kit part number. If the barcode is invalid, "Invalid barcode" displays.
Note: If the Sequencing or Binding kit selected is **incompatible**, an error message displays indicating the obsolete chemistry, and the run is **prevented** from proceeding.
- **DNA Control Complex:** PacBio **highly** recommends using the Internal Control to help distinguish between sample quality and instrument issues in the event of suboptimal sequencing performance. (**Note:** PacBio **requires** the use of the Internal Control for consumables to be eligible for reimbursement consideration.)
- **Movie time per SMRT Cell (hours):** Enter a time between 0.5 and 30. **Note:** The **SMRT Cell 8M** part supports **all** movie times up to 30 hours.
- **Use Pre-Extension:** If selected, optionally specify the length of pre-extension time in hours. This initiates the sequencing reaction prior to data acquisition. After the specified time, the sequencing reagents are removed from the SMRT Cell and replenished with fresh reagents, and data acquisition starts. This feature is useful for short inserts (such as ≤ 15 kb) and provides a significant increase in read length.
- **Include 5mC Calls in CpG Motifs:** If selected, analyzes the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.
- **Detect and Resolve Heteroduplex Reads:** Heteroduplexes are DNA molecules where the forward and reverse strands are not perfect reverse-complements. If the option is selected and heteroduplexes are detected, a consensus is called for **each** strand separately, and

the sequence of **both** strands is output.

Note: This option displays **only** if **Adeno-Associated Virus, Full-Length 16S rRNA Sequencing, <3kb Amplicons**, or **≥3kb Amplicons** are selected as the application.

Advanced options

- Specify whether to use **Adaptive Loading**. Adaptive Loading uses active monitoring of the ZMW loading process to predict a favorable loading end point. Certain steps (Cleanup and Sample Dilution) require a different buffer (Adaptive Loading Buffer) if this feature is used. **Note:** Adaptive Loading **requires** the use of Sequel® II binding kit 2.2. If you select **Yes**, fill in the following fields:
 - **Loading Target (P1 + P2):** The fraction of ZMWs that the Adaptive Loading routine will aim to load with at least one sequencing complex. The default target for CCS applications is higher to accommodate loss of complexes during pre-extension, which is generally recommended for all CCS applications.
 - **Maximum Loading Time (hours):** This defines the maximum time the system will allow loading to progress before proceeding to sequencing. (Loading time in Adaptive Loading is flexible.)
- Specify the length of time (1, 2 or 4 hours) for **immobilization** of SMRTbell templates. This is the length of time the SMRT Cell is at the Cell Prep Station to allow diffusion of SMRTbell templates into the ZMWs. This option is **not** available if **Adaptive Loading** is selected.
 - PacBio **highly recommends** using the default immobilization time of 2 hours.
- **(Sequel II systems only)** Specify, for this Run Design **only**, whether to include kinetics information (used for epigenetics analysis) in the CCS analysis output. This setting **overwrites** the global setting in **Gear > Configure > CCS Analysis Output**. **Note:** Adding kinetics information can increase the amount of storage used by the output BAM files by up to **5 times**.
- Specify, for this Run Design **only**, whether to include **low quality reads** (non-HiFi reads) in the CCS analysis output. Note that this option **disables** automatic demultiplexing, 5mC detection, and heteroduplex insert detection, if applicable.
- **Add Data to Project:** Specify that Data Sets generated by SMRT Cell(s) using this Run Design be associated with the selected Project. (This also applies to any Data Sets generated using Auto Analysis. By default, **all** Data Sets are assigned to **General Project**, which is accessible to all users.)

Editing or deleting Run Designs

1. On the home page, select **Run Design**.
2. Click the name of the Run Design to edit or delete.
3. **(Optional)** Click **View Summary** to view a table summarizing the entire Run Design.
4. **(Optional)** Click **Delete** to delete the current Run Design.
5. **(Optional)** Edit any of the fields.
6. Click **Save**.

Creating a Run Design by importing a CSV file

On a remote workstation, open the sample CSV file included with the installation.

To obtain the sample CSV files

1. On the home page, select **Run Design**.
2. Click **Import Run Design**.
3. Click **Download Template**. The ZIP file containing templates (one for Sequel II systems, and one for Sequel IIe systems) downloads to your local machine.

To update and import the CSV file

1. Update the appropriate CSV file as necessary for the Run Design. (See the definitions of the Run Design attributes in the table below.)
2. Save the edited CSV file.
3. Import the file into **Sequel ICS** using SMRT Link. To do so, first access SMRT Link using the Chrome web browser.
4. Select **Run Design**.
5. Click **Import Run Design**.
6. Select the saved CSV file designed for the run and click **Open**. The file is now imported and available for selection in Sequel ICS on the instrument.

CSV file structure

- Each CSV file row represents **one sample**. (**Note:** The Run Design CSV has a limit of **15,000 samples**.)
- The first row contains run-level information such as Run Name, Run Comments, and so on.
- For demultiplexed samples **only, one additional row** per barcode/Bio Sample Name combination is added below the master sample row.

Note: Specifying cluster settings configuration is **not yet** supported from the Run Design CSV.

Outputting Subreads on the Sequel IIe system

The Sequel IIe system can be configured to output Subreads data in BAM format by using the Run Design CSV import mechanism. In addition to the other required columns, users can add the column **Emitted Subreads Percent** to the CSV file, with a value of 0-100 for a given collection. This results in the inclusion of Subreads from 0-100% of ZMWs in the Data Set transferred from the instrument, in a BAM file **separate** from the HiFi

reads. Note that this will **not** result in the inclusion of associated scraps data for each ZMW.

Run Design attribute	Required	Description
Experiment Name	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Defaults to Run Name. Example: Standard_Edna.1
Experiment Id	No	Enter a valid experiment ID. Example: 325/3250057 <ul style="list-style-type: none"> Experiment IDs cannot contain the following characters: <code><, >, :, ", \, , ?, *, or)</code>. Experiment IDs cannot start or end with a <code>/</code> and cannot have two adjacent <code>/</code> characters, such as <code>//</code>. Experiment IDs cannot contain spaces. Specifically, Experiment IDs cannot satisfy the regular expressions: <code>/[<>:"\\ ?*]/g</code>, <code>/(\?:^\\)\\\/ (\?:\/\$)/</code>, <code>/ /g</code>
Experiment Description	No	Enter any ASCII string. Defaults to Run Comments. Example: 20170530_A6_VVnC_SampleSheet
Run Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Run name must be entered for the first cell and will be applied to the remaining cells in the run. Example: 20170530_A6_VVnC_SampleSheet
System Name	No	Must be <code>Sequel II</code> or <code>Sequel IIE</code> .
Run Comments	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: <code>ecoliK12_March2021</code>
Is Collection	No	Enter a Boolean value. (See Boolean details below.) Specifies whether the row designates a Collection (<code>TRUE</code>) or a barcoded sample (<code>FALSE</code>). <ul style="list-style-type: none"> Collection lines should have the Barcode Name and Bio Sample Name fields blank. Barcoded Sample lines only need to include the Is Collection, Sample Name, Barcode Name, and Bio Sample Name fields.
Sample Well	Yes	Must be specified in every row. Well number must start with a letter <code>A</code> through <code>H</code> , and end in a number <code>01</code> through <code>12</code> , i.e. <code>A01</code> through <code>H12</code> . It must satisfy the regular expression <code>``/^[A-H](?:0[1-9] 1[0-2])\$/``</code> Example: <code>A01</code>
Well Sample Name	Yes	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: <code>A6_3230046_A01_SB_ChemKitv2_8rxnKit</code> Note: The Sample Name must be unique within a run.
Movie Time per SMRT Cell (hours)	Yes	Enter a floating point number between 0.1 and 30. Time is in hours. Example: 5
Use Adaptive Loading	No	Enter a Boolean value. (See Boolean details below.)
Loading Target (P1 + P2)	No	Enter a floating point number between 0.01 and 1. Example: 0.4
Maximum Loading Time (hours)	No	Enter a floating point number between 1 and 2. Time is in hours. Example: 1.2
Sample Comment	No	Enter alphanumeric characters, spaces, hyphens, underscores, colons, or periods only. Example: <code>A6_3230046_A01_SB_BindKit_ChemKit</code>
Insert Size (bp)	Yes	Enter an integer ≥ 10 . Units are in base pairs. Example: 2000
On Plate Loading Concentration (pM)	No	Enter a floating point number. Units are in parts per million. Example: 5

Run Design attribute	Required	Description
Size Selection	No	Enter a Boolean value. (See Boolean details below.) Default is <code>FALSE</code> .
Template Prep Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1117100259100111716
DNA Control Complex Box Barcode	No	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1234101084300123120
Binding Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM1117100862200111716
Sequencing Kit Box Barcode	Yes	Enter or scan a valid kit barcode. (See Kit Barcode Requirements details below.) Working example: DM0001100861800123120
Automation Name	No	Enter <code>diffusion</code> (not case-sensitive) or a custom script. (Sequel II systems do not support magbead loading.) A path can also be used, such as <code>/path/to/my/script/my_script.py</code> . The path will not be processed further, so if the full URI is required, it must be included in the CSV, such as <code>chemistry://path/to/my/script/my_script.py</code> .
Automation Parameters	No	To enable Pre-Extension time, enter the number of hours and set the boolean value to <code>TRUE</code> . Example 2 hours: <code>ExtensionTime=double:2 ExtendFirst=boolean:TRUE</code> (Note: Leave blank when not using Pre-Extension time, or set the boolean value to <code>FALSE</code> .)
Detect and Resolve Heteroduplex Reads	No	Enter a boolean value. (See Boolean details below.) Set to <code>TRUE</code> to allow for detection of heteroduplex reads. Note: Only applicable if Application is set to one of the following: <ul style="list-style-type: none"> • Adeno-Associated Virus • <3kb Amplicons • >=3kb Amplicons • Custom
Include 5mC Calls in CpG Motifs	No	Enter a boolean value. (See Boolean details below.) Set to <code>TRUE</code> to allow for 5mC calls in CpG motifs. Note: Only applicable if Application is set to <code>HiFi Reads</code> or <code>Custom</code> .
Sample is Barcoded	No	Enter a boolean value. (See Boolean details below.) Set to <code>TRUE</code> for a barcoded run.
Demultiplex Barcodes	No	Add any of the following values: <code>Do Not Generate</code> , <code>In SMRT Link</code> , or <code>On Instrument</code> . If left blank, the default is <code>Do Not Generate</code> for all systems. Note: This is available for all applications. The following values are recommended based upon your system: <ul style="list-style-type: none"> • Sequel II system: Enter one of the following values: <code>Do Not Generate</code> or <code>In SMRT Link</code>. • Sequel IIe system: Enter one of the following values: <code>Do Not Generate</code>, <code>In SMRT Link</code>, or <code>On Instrument</code>.
CCS Analysis Output - Include Low Quality Reads	No	Enter a boolean value. (See Boolean details below.) <ul style="list-style-type: none"> • Set to <code>TRUE</code> to allow for CCS analysis with <code>--all mode</code> activated and produce a <code>reads.bam</code> file • Set to <code>FALSE</code> to exclude all reads with <code>rq < 0.99</code>.

Run Design attribute	Required	Description
Barcode Set	No	Must be a UUID for a Barcode Set present in the database. To find the UUID: Click Data Management > View Data > Barcodes . Click the Barcode file of interest, then view the UUID. Example: dad4949d-f637-0979-b5d1-9777eff62008 Note: This field is used for demultiplexed data.
Same Barcodes on Both Ends of Sequence	No	Enter a boolean value. (See Boolean details below.) Set to TRUE if symmetric, FALSE if asymmetric.
Barcode Name	No	Enter Barcode Names one per line. Example: bc1001--bc1001 <ul style="list-style-type: none"> • Use double hyphens (--) to separate the 2 barcodes of each pair. • The barcode names must be contained within the specified Barcode Set. • A given barcode name cannot appear more than once in the spreadsheet. • A maximum of 15,000 barcodes is permitted per sample.
Bio Sample Name	Yes	Enter Bio Sample Names in the same row as their associated Barcode Names. Use alphanumeric characters, spaces (allowed but not recommended for compatibility with downstream software), hyphens, underscores, colons, or periods only . Bio Sample Names cannot be longer than 40 characters. Example: sample1 Note: This field is used for collections for non-multiplexed data, and for barcoded samples in multiplexed data.
Pipeline ID	No	Note: This field is required to create an Auto Analysis. <ul style="list-style-type: none"> • 5mC CpG Detection: cromwell.workflows.pb_detect_methyl • Demultiplex Barcodes: cromwell.workflows.pb_demux_ccs • Export Reads: cromwell.workflows.pb_export_ccs • Genome Assembly: cromwell.workflows.pb_assembly_hifi • HiFi Mapping: cromwell.workflows.pb_align_ccs • HiFiViral SARS CoV-2 Analysis: cromwell.workflows.pb_sars_cov2_kit • Iso-Seq Analysis: cromwell.workflows.pb_isoseq3_ccsonly • Mark PCR Duplicates: cromwell.workflows.pb_mark_duplicates • Microbial Genome Analysis: cromwell.workflows.pb_microbial_analysis • Read Segmentation: cromwell.workflows.pb_segment_reads • Read Segmentation and Single-Cell Iso-Seq Analysis: cromwell.workflows.pb_segment_reads_and_sc_isoseq • Single-Cell Iso-Seq Analysis: cromwell.workflows.pb_sc_isoseq • Structural Variant Calling: cromwell.workflows.pb_sv_ccs • Trim Ultra-Low Adapters: cromwell.workflows.pb_trim_adapters
Analysis Name	No	Enter any ASCII string. See Auto Analysis Fields below for details. Note: This field is required for Auto Analysis, otherwise the name will be "". Example: sample 1 analysis

Run Design attribute	Required	Description
Entry Points	No	<p>Entry Points only apply to Barcode Sets and Reference Sets. In addition, this field is required for Auto Analysis.</p> <p>Enter an ASCII string in the format <code>file_type;entry_id;uuid</code>, with parameters separated by characters.</p> <ul style="list-style-type: none"> To find the UUID: Click Data Management > View Data > HiFi Reads or Subreads. Click the Data Set of interest, then view the UUID. See the SMRT® Tools reference guide section Appendix A - Application entry points and output files to see the entry point names for each application. <p>Example: PacBio.DataSet.BarcodeSet;eid_barcode;afe89e3f-17ca-e9b8-eae9-b701dbb1f02d PacBio.DataSet.ReferenceSet;eid_ref_dataset;6b8db144-a601-4577-ab04-ba64cad0548</p>
Task Options	No	<p>Enter an ASCII string containing the options for the application referred to in the Pipeline ID field, with parameters separated by “;” characters: <code>task_id;value_type;value</code>.</p> <p>Example: pbmm2_align.task_options.minalnlength;integer;50</p> <p>Note: This field is optional for Auto Analysis - any task options not specified will use pipeline defaults.</p>
Application	No	<ul style="list-style-type: none"> HiFi Reads Microbial Assembly Iso-Seq Method MAS-Seq single cell HiFiViral SARS-CoV-2 Adeno-Associated Virus Full-Length 16S rRNA Sequencing Shotgun Metagenomic Profiling or Assembly <3kb Amplicon Sequencing ≥3kb Amplicon Sequencing Custom <p>If blank or contains invalid values, default is Custom.</p>
CCS Analysis Output - Include Kinetics Information	No	<p>Enter a boolean value. (See Boolean details below.) Set to TRUE to specify that CCS analysis output includes kinetics information (used for epigenetics analysis.) Note: Adding kinetics information can increase the amount of storage used by the output BAM files by up to 5 times.</p>

CSV file general requirements

- Each line in the CSV file represents **one** sample.
- The CSV file may **only** contain ASCII characters. Specifically, it must satisfy the regular expression `/^[\\x00-\\x7F]*$/g`

Boolean values

- Valid boolean values for **true** are: `true`, `t`, `yes`, or `y`.
- Valid boolean values for **false** are: `false`, `f`, `no`, or `n`.
- Boolean values are **not** case-sensitive.

Kit barcode requirements

Kit barcodes are composed of three parts used to make a single string:

1. Lot Number (Example: DM1234)
2. Part Number (Example: 100-619-300)
3. Expiration Date (Example: 2020-12-31)

For the above example, the full kit barcode would be:

DM1234100619300123120.

Each kit **must** have a valid Part Number and **cannot** be obsolete. The list of kits can be found through a services endpoint such as:

```
[server name]:[services port number]/smrt-link/bundles/chemistry-pb/active/
files/definitions%2FPacBioAutomationConstraints.xml
```

This services endpoint will list, for each kit, the part numbers (PartNumber) and whether it is obsolete (IsObsolete).

Dates must also be valid, meaning they must exist in the Gregorian calendar.

Auto Analysis fields

- The fields include Pipeline ID, Analysis Name, Entry Points, and Task Options.
- You can define **one** analysis for each Collection or Bio Sample. The Pipeline ID, Analysis Name and Entry Points fields are **required** to create an Auto Analysis.
- The analysis name is a concatenation of the values of the **Analysis Name** and **Bio Sample Name** fields.
- The **Task Options** field may be left blank; any task options not specified will use pipeline defaults.

Run QC

Use SMRT Link's **Run QC** module to monitor performance trends and perform run QC remotely.

Metrics can be reviewed in the Run QC module. **All** Sequel II systems and Sequel IIe systems connected to SMRT Link can be reviewed using Run QC.



1. Access SMRT Link using the Chrome web browser.
2. Select **Run QC**.

Accessing instrument status

Instrument Name ↑	Instrument Status	SMRT Cell Status	Run Completion	Sequencing ZMWs
64002e	Ready	■ ■	Completed 2 hours ago on 2022-03-08, 01:47:44 PM.	3,500,000 3,000,000 2,000,000 1,500,000 3,000,000 2,000,000 3,500,000
64009e	Ready	■ ■ ■ ■	Completed 1 day and 12 hours ago on 2022-03-07, 04:09:07 AM.	3,500,000 3,000,000 2,500,000 2,000,000 3,500,000
64001e	Running	■ ■ ■ ■	In 2 days and 8 hours at 04:13:44 AM.	3,000,000 2,500,000 3,200,000 3,000,000 2,800,000 2,600,000 3,200,000
64263e	Running	■	In 8.4 hours at 12:16:51 AM.	3,000,000 2,800,000 2,600,000 3,200,000
64012e	Ready ▲	■	Completed 2.7 hours ago on 2022-03-08, 01:05:21 PM.	3,000,000 2,800,000 2,600,000

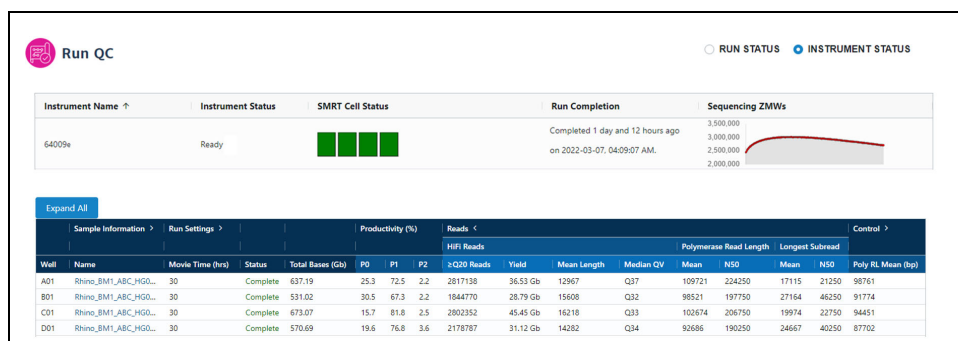
Last Update: 2022-03-08, 03:36:57 PM

1. Select **Instrument Status**. For **each** instrument connected to the instance of SMRT Link, this displays the instrument name and its current status, SMRT Cell status, when the run will be completed, any active alarms, and how many sequencing ZMWs are active.
 - A **red alarm** symbol displays next to the instrument status if any errors or warnings appear during a sequencing run.
 - If an instrument does **not** have a SMRT Cell tray loaded, the SMRT Cell Status field will **not** display any icons. The icons are:
 - **Fully green**: The SMRT Cell has completed sequencing.
 - **Half green**: The SMRT Cell is being prepared or is currently sequencing.
 - **White**: The SMRT Cell is in the queue for sequencing, but cell preparation has not started.

- **Run Completion:** Displays the estimated time remaining to complete sequencing run or the time elapsed since the sequencing run completed. Also displays the date (in YYYY-MM-DD format) when the last sequencing run was completed.
- **Sequencing ZMWs:** Displays a plot of how many ZMWs on a SMRT Cell are actively sequencing during a movie collection. For sequencing runs conducted with Binding kit 2.2 and 3.2, **only** the number of actively sequencing singly-loaded ZMWs (P1) displays. For sequencing runs conducted with Binding kit 2.1 and 3.1, the **total** number of actively sequencing ZMWs (P1 + P2) displays.

Note: Due to terminations, not **all** ZMWs are singly-loaded at the same time. Some ZMWs are singly-loaded only **at or near the end** of a movie collection, whereas others are singly-loaded only at the **beginning**. (**Singly-loaded** means that the ZMW contains only one active polymerase instead of two or more simultaneously active polymerases.) For runs conducted with Binding kit 2.2 and 3.2, the peak concurrent **Sequencing ZMWs** value shown in the plot will always be **less** than the final %P1 ZMW yield reported in the Run QC metrics table at the end of a movie collection. (For sequencing runs conducted with Binding kit 2.1 and 3.1, the peak concurrent Sequencing ZMWs value shown in the plot will always be **higher** than the final %P1 ZMW yield reported in Run QC.) For a SMRT Cell that achieves $\geq 50\%$ P1 loading and $\geq 10\%$ P0, the ZMW Sequencing plot should typically display a peak value above 2,000,000.

See the figure below for an example comparison between the **Instrument Status report** (top) and **Run QC report** (bottom) for a WGS sample sequenced using Binding kit 3.2 with a 30-hour movie collection time. The Sequencing ZMWs plot in the Instrument Status report shows that the peak concurrent Sequencing ZMWs value for the last SMRT Cell in the run (Well D01) is approximately 3,000,000 ZMWs, whereas the final %P1 ZMW yield reported in the corresponding Run QC metrics table for Well D01 is 76.8% (or 6,144,000 P1 ZMWs.)



Accessing run information

Run QC

Sequel II Sequel IIE

Displaying rows 1 to 3 out of 3

		Dates					Instrument Details	
<input type="checkbox"/>	Name	Summary	Run Date	Completion D...	Transferred D...	Created By	Status	Instrument Na...
<input type="checkbox"/>	20210925_64011_Quokka...	20210925_64011_Quokka_JW_ML...	2021-09-25, 09:47:0...	2021-09-27, 02:22:...	2021-09-27, 05:13:...	Jellison	Complete	64011
<input type="checkbox"/>	Redwood_Bulk_5_LZ		2020-02-13, 07:05:5...	2020-02-17, 09:41:...	2020-02-19, 09:56:...	lthru	Complete	Sequel II Instrument
<input type="checkbox"/>	2018-07-27_64003_30kEco...	Ecoli	2018-07-27, 07:46:3...	2018-07-28, 07:30:...	2018-07-29, 06:57:...	rdalal	Complete	Sequel

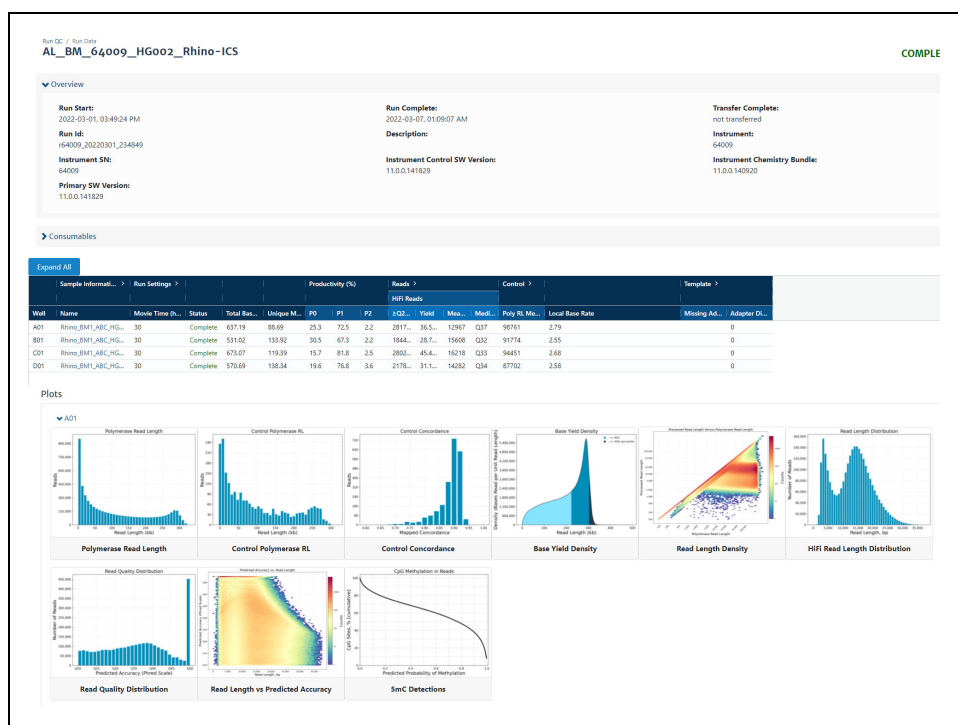
1. Select **Run Status**.
2. Click **Sequel II** or **Sequel IIE** to view informations on runs for a specific instrument model.
3. Runs can be sorted and searched for:
 - To sort runs, click a **column title**.
 - To search for a run, enter a unique search string into the **Search** field.
4. Click the run to view information about that run.
5. To **export** Run QC data in CSV format: Select one or more runs in the table, then click **Export Selected**.

Table fields

Note: Not all table fields are shown by default. To see **additional** table fields, click the > symbol next to a column title.

- **Name:** A list of all runs for the instruments connected to SMRT Link. Click a run name to view more detailed information on the individual run page.
- **Summary:** A description of the run.
- **Dates**
 - **Run Date:** The date and time when the run was started.
 - **Completion Date:** The date and time the run was completed.
 - **Transferred Date:** The date and time the run results were transferred to the network.
- **Created By:** The name of the user who created the run.
- **Status:** The current status of the run. Can be one of the following: **Running, Complete, Failed, Terminated, or Unknown**.
- **Instrument Details**
 - **Instrument Name:** The name of the instrument.
 - **Instrument SN:** The serial number of the instrument.
 - **Instrument SW:** The versions of Sequel Instrument Control Software (ICS) installed on the instrument.
- **Cells**
 - **Total:** The total number of SMRT Cells used in the run.
 - **Completed:** The number of SMRT Cells that generated data for the run.

- **Failed:** The number of SMRT Cells that failed to generate data during the run.
 - **Run ID:** An internally-generated ID number identifying the run.
 - **Primary Analysis SW:** The version of Primary Analysis software installed on the instrument.
 - **UUID:** Another internally-generated ID number identifying the run.
6. Click the **Run name** of interest. Following are the fields and metrics displayed.



- **Run Start:** The date and time when the run was started.
 - **Run Complete:** The date and time the run was completed.
 - **Transfer Complete:** The date and time that the run data was successfully transferred to the network.
 - **Run ID:** An internally-generated ID number identifying the run.
 - **Description:** The description, as defined when creating the run.
 - **Instrument:** The name of the instrument.
 - **Instrument SN:** The serial number of the instrument.
 - **Instrument Control SW Version:** The versions of Sequel Instrument Control Software (ICS) installed on the instrument.
 - **Instrument Chemistry Bundle:** The version of the Chemistry Bundle installed on the instrument when the run was initiated.
 - **Primary SW Version:** The versions of Primary Analysis software installed on the instrument.
7. Click the **>** arrow at the top of the **Consumables** table to see the sample wells used, consumable type, lot number, expiration date, and other information.

Run settings and metrics

Note: Click **Expand All** to expand all of the table columns. Click **Collapse All** to collapse the table columns.

- **Well:** The ID of an individual well used for this sample.
- **Sample Information**
 - **Name:** The sample name, as defined when creating the run. Clicking the name will take you to the corresponding entry in the **Data Management** module.
 - **Comment:** Sample comment, entered in Run Design.
- **Run Settings**
 - **Movie Time (hrs):** The length of the movie associated with this SMRT Cell.
 - **Loading Concentration (pM):** The on-plate loading concentration, in picomolarity.
 - **Pre-extension Time (hrs):** The pre-extension time used in the collection, if any.
 - **Workflow:** The instrument robotics workflow used for the run.
 - **Loading Time:** The time the system took for loading to progress before proceeding to sequencing.
- **Status:** The current collection status for the SMRT Cell. This can be one of the following: **Complete, Collecting, Aborted, Failed, In Progress, or Pending.**
- **Total Bases (Gb):** Calculated by multiplying the number of **productive** (P1) ZMWs by the mean polymerase read length; displayed in Gigabases.
- **Unique Molecular Yield (Gb):** The sum total length of unique single molecules that were sequenced. It is calculated as the sum of per-ZMW median subread lengths.
- **Productivity (%)**
 - **P0:** Empty ZMW; no signal detected.
 - **P1:** ZMW with a high quality read detected.
 - **P2:** Other, signal detected but no high quality read.
- **Reads:** Polymerase reads are trimmed to the high-quality region and include bases from adapters, as well as potentially multiple passes around a SMRTbell template.
 - **HiFi Reads ≥Q20 Reads:** The total number of CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Yield:** The total yield (in base pairs) of the CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Mean Length:** The mean read length of the CCS reads whose quality value is equal to or greater than 20.
 - **HiFi Reads Median QV:** The median number of CCS reads whose quality value is equal to or greater than 20.
 - **Polymerase Read Length Mean:** The mean high-quality read length of all polymerase reads. The value includes bases from adapters as well as multiple passes around a circular template.

- **Polymerase Read Length N50:** 50% of all read bases came from polymerase reads longer than this value.
 - **Longest Subread Mean:** The mean subread length, considering only the longest subread from each ZMW.
 - **Longest Subread N50:** 50% of all read bases came from subreads longer than this value when considering only the longest subread from each ZMW.
 - **Control**
 - **Poly RL Mean (bp):** The mean polymerase read length of the control reads.
 - **Total Reads:** The number of control reads obtained.
 - **Concordance Mean:** The average concordance (agreement) between the control raw reads and the control reference sequence.
 - **Concordance Mode:** The median concordance (agreement) between the control raw reads and the control reference sequence.
 - **Local Base Rate:** The average base incorporation rate, excluding polymerase pausing events.
 - **Template**
 - **Missing Adapter (%):** The percent of pre-filter ZMWs that are missing adapters.
 - **Adapter Dimer:** The percent of pre-filter ZMWs which have observed inserts of 0-10 bp. These are likely adapter dimers.
 - **Short Insert:** The percent of pre-filter ZMWs which have observed inserts of 11-100 bp. These are likely short fragment contamination.
8. View plots for each SMRT Cell where data was successfully transferred. Clicking on an individual plot displays an expanded view. These plots include:
- **Polymerase Read Length:** Plots the number of reads against the polymerase read length.
 - **Control Polymerase RL:** Displays the polymerase read length distribution of the control, if used.
 - **Control Concordance:** Maps control reads against the known control reference and reports the concordance.
 - **Base Yield Density:** Displays the number of bases sequenced in the collection, according to the length of the read in which they were observed. Values displayed are per unit of read length (i.e. the base yield density) and are averaged over 2000 bp windows to gently smooth the data. Regions of the graph corresponding to bases found in reads longer than the N50 and N95 values are shaded in medium and dark blue, respectively.
 - **Read Length Density:** Displays a density plot of reads, hexagonally binned according to their high-quality read length and median subread length. For very large insert libraries, most reads consist of a single subread and will fall along the diagonal. For shorter inserts, subreads

will be shorter than the HQ read length, and will appear as horizontal features. This plot is useful for quickly visualizing aspects of library quality, including insert size distributions, reads terminating at adapters, and missing adapters.

- **HiFi Read Length Distribution:** Displays a histogram distribution of HiFi reads ($QV \geq 20$), other CCS reads (three or more passes, but $QV < 20$), and other reads, by read length.
- **Read Quality Distribution:** Displays a histogram distribution of HiFi reads ($QV \geq 20$) and other CCS reads by read quality.
- **Read Length vs Predicted Accuracy:** Displays a heat map of CCS read lengths and predicted accuracies. The boundary between HiFi reads and other CCS reads is shown as a dashed line at $QV 20$.
- **5mC Detections:** If 5mC calling in CpG motifs was performed, this plot displays a reverse cumulative distribution of all detected CpG motifs according to their predicted probability of methylation.

Data Management

Use the **Data Management** module to:

- Create and manage Data Sets,
- View Data Set information,
- Create and manage Projects,
- View, import, export, or delete sequence, reference, and barcode data.

What is a Data Set?

Data Sets are logical collections of sequencing data (basecalled or analyzed) that are analyzed together, and for which reports are created. Data Sets:

- Help to **organize** and **manage** basecalled and analyzed data. This is especially valuable when dealing with large amounts of data collected from different sequencing runs from one or more instruments.
- Are the way that sequence data is represented and manipulated in SMRT Link. Sequence data from the instrument is organized in Data Sets. Data from **each** cell or collection is a Data Set.
- Can be used to collect data and summarize performance characteristics, such as data throughput, while an experiment is in progress.
- Can be used to generate reports about data, and to exchange reports with collaborators and customers.
- Can be used to start a job. (See [“Starting a job from a Data Set” on page 37](#) for details.)

A Data Set can contain sequencing data from **one** or **multiple** SMRT Cells or collections from different runs, or a portion of a collection with multiplexed samples.

For more information on Data Sets, click [here](#).

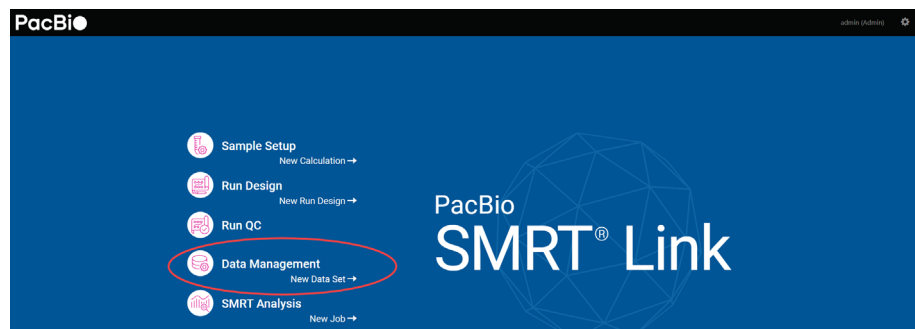
In SMRT Link, movies, cells/collections, context names and well samples are all in one-to-one relationships and can be used more or less interchangeably. That is, a Data Set from a single cell or collection will also be from a single collection derived from DNA from a single well sample. Data produced by SMRT Cells, however, can be used by **multiple** Data Sets, so that data may have a many-to-one relationship with collections.

Some Data Sets can contain **basecalled** data, while others can contain **analyzed** data:

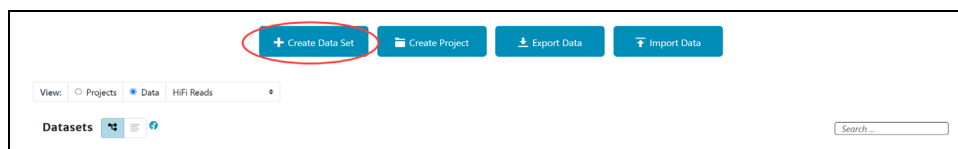
- **Basecalled data** Data Sets contain sequence data from one or multiple cells or collections.
- **Analyzed data** Data Sets contain data from previous analyse(s).

Elements within a Data Set are of the same data type, typically subreads or consensus reads, in aligned or unaligned format.

Creating a Data Set



1. Access SMRT Link using the Chrome web browser.
2. Select **Data Management**.
3. Data Sets can be sorted and searched for:
 - To sort Data Sets, click a **column title**.
 - To search for a Data Set, use the Search function. See [“Appendix B - Data search” on page 147](#) for details.



4. Click **+ Create Data Set**.
5. Enter a name for the new Data Set.

Create Data Set

Cancel Save Data Set

Data Set Name Required: Data_Set_98

Well Sample Name Required: SMS_Kiwi_PA_S4006_3280182_A01_lambdaBsaA1_Diffusion_0.75pM

Bio Sample Name Required: SMS_Kiwi_PA_S4006_3280182_A01_lambdaBsaA1_Diffusion_0.75pM

Filter Reads by QV ≥ ☐ ☐

Filter Reads by Length bp to bp

Data Type: HiFi Reads

Datasets

Data Set Details				Sample Details			Run Data
Name	Well Sample Name	Run Name	Date Created	Created By	Bio Sample Name	Barcode Name	Total Length of Read
Tiny Lambda m54006...	SMS_Kiwi_PA_S4006_3...	20180706_A6_Kiwi...	2021-10-06, 11:28...	smrtlinktest	SMS_Kiwi_PA_S4006_3...		284,298

6. Select the type of data to include in the new Data Set:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
7. (Optional) Specify the **Project** that this new Data Set will be associated with using the **Projects** menu (located at the top-right of

the Data Management page.) **General Project:** This Data Set will be visible to **all** SMRT Link users. **All My Projects:** This Data Set will be visible **only** to users who have access to Projects that you are a member of.

Note: Selecting a Project **also** filters the Data Sets that you can use when **creating** the new Data Set.

8. In the **Data Sets** table, select one or more sets of sequence data.
9. **(Optional)** Choose how to **view** the Data Set table: 1) Tree Mode - A barcoded Data Set displays as **one row**. 2) Flat Mode - A barcoded Data Set and its demultiplexed subsets display as **separate rows**.
10. **(Optional)** Use the Search function to search for specific Data Sets. See ["Appendix B - Data search" on page 147](#) for details.
11. **(Optional)** If you selected **one** Data Set **only**, click the **Filter Reads by QV** box above the Data Set list. Enter the minimum quality value to retain in the new Data Set.
12. **(Optional)** If you selected **one** Data Set **only**, click the **Filter Reads by Length** box above the Data Set list. Enter the minimum and/or maximum length to retain in the new Data Set.
13. Click **Save Data Set**. The new Data Set becomes available for starting analyses, viewing, or generating reports.
14. After the Data Set is created, click its name in the main Data Management screen to see reports, metrics, and charts describing the data included in the Data Set. See ["Data Set QC reports" on page 37](#) for details.



Viewing Data Set information

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of Data Set to view:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See ["Appendix B - Data search" on page 147](#) for details.
4. Click the name of the Data Set to see information about the sequence data included in the Data Set, as well as QC reports.

Copying a Data Set

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of data to copy:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.

- **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.

3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 147](#) for details.
4. Click the name of the Data Set to copy. The Data Set Reports page displays.
5. Click **Copy**. The main Data Management page displays; the new Data Set has **(copy)** appended to the name.

Deleting a Data Set

Note: SMRT Link's Delete Data Set functionality deletes the Data Set from the SMRT Link interface **only**, **not** from your server.

It is good practice to export Data Sets you no longer need to a backup server, then delete them from SMRT Link. This frees up space in the SMRT Link interface.

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of data to delete:
 - **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
- The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 147](#) for details.
4. Click the name of the Data Set to delete.
5. Click **Delete**. Note that this deletes the Data Set from the SMRT Link interface **only**; **not** from your server. To delete the Data Set from your server, **manually** delete it from the disk.
6. Click **Yes**. The Data Set is no longer available from SMRT Link.

Starting a job from a Data Set

From the Data Set reports page, a job can be started using the Data Set.

1. Click **Analyze...**, then name the job.
2. Select **Analysis** or **Utility** as the workflow type, then click **Next**.
3. Follow the instructions starting at Step 12 of [“Creating and starting a job” on page 44](#).

Data Set QC reports

The Data Set QC reports are generated when you create a new Data Set or update the data contained in existing Data Sets. These reports are designed to provide all relevant information about the data included in the Data Set as it comes from the instrument prior to data analysis, and are useful for data QC purposes.

The following reports are generated by default:

The screenshot shows a web interface for data management. On the left, there is a sidebar with a 'Dataset Overview' section containing links for 'Status', 'Thumbnails', 'Display All', 'CCS Analysis Report', 'Analyses', and 'Data'. The main area is titled 'Status' and displays detailed information about the dataset 'Tiny Kiwi ecoli 3-plex (CCS)'. The information includes:

- Dataset:** Tiny Kiwi ecoli 3-plex (CCS)
- Data Set ID:** 28
- Data Set UUID:** a3180f93-496b-a3ed-1c21-3aacc34d3b75
- Well Sample Name:** SMS_Kiwi_Verif_54043_3280212_A01_VVCT158_28_384plex_jobx40_Diffusion_2.5pM
- Biological Sample Name:** [multiple]
- Description:** ccs dataset converted
- Number of Records:** 20
- Total Length:** 47,113
- Status:** SUCCESSFUL
- Date Created:** 2018-10-01, 10:51:33 AM
- Date Imported:** 2020-09-24, 10:13:54 AM
- Date Updated:** 2020-09-24, 10:15:00 AM
- Job ID:** 39
- Data Path:** /jpb/dept/secondary/bv/testdata/SAS-Sequel/ecoli/54043_20180917_210804/1_A01_micro/m54043_180917_211649_micro.consensusreadset.xml
- Run name:** 20180917_43_Kiwi_Verif
- Cell Index:** 0
- Cell Id:** BA343109
- Instrument Name:** Flex43
- Well Name:** A01
- Metadata Context Id:** m54043_180917_211649

Data Set Overview > Status

Displays the following information about the Data Set:

- The Data Set Name, ID, description, and when it was created and updated.
- The number of reads and their total length in base pairs.
- The names of the run and instrument that generated the data.
- The biological sample name and well sample names of the sample used to generate the data.
- Path to the location on your cluster where the data is stored, which can be used for command-line navigation. For information on command-line usage, see **SMRT® Tools reference guide (v11.1)**.

Completed Analyses

Lists all completed analyses that used the Data Set as input. To view details about a specific analysis, click its name.

Raw Data Report > Summary Metrics

- **Polymerase Read Bases:** The total number of polymerase read bases in the Data Set.
- **Polymerase Reads:** The total number of polymerase reads in the Data Set.
- **Polymerase Read Length (mean):** The mean read length of all polymerase reads in the Data Set.
- **Polymerase Read N50:** The read length at which 50% of all the bases in the Data Set are in polymerase reads longer than, or equal to, this value.
- **Subread Length (mean):** The mean read length of all subreads in the Data Set.
- **Subread N50:** The length at which 50% of all the subreads in the Data Set are longer than, or equal to, this value.
- **Insert Length (mean):** The mean length of all the inserts in the Data Set.
- **Insert N50:** The length at which 50% of all the inserts in the Data Set are longer than, or equal to, this value.

Information on loading, control reads, and adapters is also displayed. Other information may display based on the Data Set type.

What is a Project?

- Projects are collections of Data Sets, and can be used to restrict access to Data Sets to a subset of SMRT Link users.
- By default, **all** Data Sets and data belong to the **General Project** and are accessible to **all** users of SMRT Link.
- **Any** SMRT Link user can create a Project and be the owner. Projects must have an owner, and can have **multiple** owners.
- Unless a Project is shared with other SMRT Link users, it is **only** accessible by the owner.
- Only owner(s) can delete a Project; deleting a Project deletes **all** Data Sets and analyses that are part of the Project.

Projects include:

- One or more Data Sets and associated Quality Control information.
- One or more analysis results and the associated Data Sets, including information for all analysis parameters and reference sequence (if used).

Data Sets and Projects

- Once created, a Data Set **always** belongs to at least **one** project; either the **General** project or another project the user has access to.
- Data Sets can be associated with **multiple** projects.
- The data represented by a Data Set can be copied into **multiple** projects using the Data Management report page **Copy** button. Any changes made to a particular copy of a Data Set affect **only** that copy, **not** any other copies in other Projects. If a Data Set is to be used with multiple Projects, PacBio recommends that you make a **separate copy** for each Project.
- Use the **Projects** menu (located at the top-right of the Data Management page) to filter the Data Sets displayed; this is based on which Projects the Data Sets are associated with.

Creating a Project

The screenshot shows the 'Create Project' form in the SMRT Link Data Management interface. The form is divided into several sections:

- Project Name:** A required text field containing 'QC_Group'.
- Description:** A text area for project details.
- Associated Data Sets:** A section with a 'Select Data Sets' button.
- Members:** A section for managing project access.
 - Access for All SMRT® Link users:** A dropdown menu set to 'None'.
 - Access for Individual SMRT® Link Users:** A list of users with 'View' buttons.

User Name	Email
Administrator (Administrator@p...	
EPMAAdmin2	
 - QA Section:** A search bar and a table for selecting individual users.

User Name	Email
<input type="checkbox"/> sappi-adm-qa	sappi-adm-qa@pacifibiosciences.com

Buttons for 'Cancel' and 'Save' are located at the top right. An 'Add Selected User' button is at the bottom of the QA section.

1. Access SMRT Link using the Chrome web browser.
2. Select **Data Management**.

3. Click **+ Create Project**.
4. Enter a name for the new project.
5. **(Optional)** Enter a description for the project.
6. Click **Select Data Sets** and select one or more sets of sequence data to associate with the project.
 - **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 147](#) for details.
7. **(Optional)** Share the Project with other SMRT Link users. **(Note:** Unless a Project is shared, it is **only** visible to the owner.) There are two ways to specify who can access the new Project, using the controls in the **Members** section:
 - **Access for all SMRT Link Users: None** - No one can access the project other than the user who created it; **View** - Everyone can view the Project; **View/Edit**: Everyone can see and edit the Project.
 - **Access for Individual SMRT Link Users**: Enter a user name and click **Search By Name**. Choose **Owner**, **View**, or **View/Edit**, then click **Add Selected User**.
 - **Notes:** A) Projects can have **multiple** owners. B) If you enable **all** SMRT Link users to have **View/Edit** access, you cannot change an individual member's access to **View**.
8. Click **Save**. The new project becomes available for SMRT Link users who now have access.

Editing a Project

1. On the home page, select **Data Management**.
2. Click **View > Projects**.
3. Projects can be sorted and searched for:
 - To sort Projects: Click a **column title**.
 - To search for a Project, use the Search function. See [“Appendix B - Data search” on page 147](#) for details.
4. Click the name of the project to edit.
 - **(Optional)** Edit the Project name or description.
 - **(Optional)** Delete a Data Set associated with the Project: Click **X**.
 - **(Optional)** Add one or more sets of sequence data to the Project: Click **Select Data Sets** and select one or more Data Sets to add.
 - **(Optional)** Delete members: Click **X** next to a Project member's name to delete that user from access to the Project.
 - **(Optional)** Add members to the Project: See Step 7 in **Creating a Project**.
5. Click **Save**. The modified Project is saved.

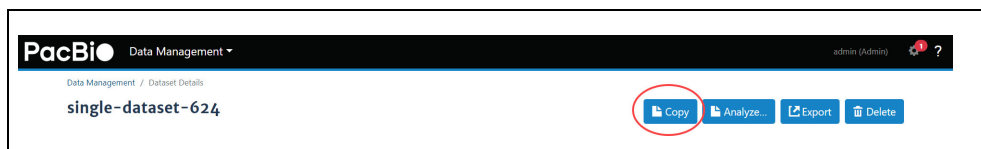
Deleting a Project

1. On the home page, select **Data Management**.
2. Click **View > Projects**.
3. Click the name of the Project to delete.
4. Click **Delete**. (This deletes **all** Data Sets and analyses that are part of the Project from SMRT Link, but **not** from the server.)

Viewing/deleting sequence, reference and barcode data

1. On the home page, select **Data Management**.
2. Click **View > Data**, then choose the type of data to view or delete:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
 - **Barcodes**: Barcodes from barcoded samples.
 - **References**: Reference sequence FASTA files used when creating certain analyses.
3. (**Optional**) Use the Search function to search for specific Data Sets, barcode files or reference sequence files. See ["Appendix B - Data search" on page 147](#) for details.
4. Click the name of the sequence, reference or barcode file of interest. Details for that sequence, reference sequence file or barcode file display.
5. (**Optional**) To delete the sequence data, reference sequence, or barcode file, click **Delete**.

Note: The **Copy** button is available for Subreads and HiFi reads, but **not** for Reference and Barcode data.



Importing sequence, reference and barcode data

Note: If your Sequel II system or Sequel IIe system is linked to the SMRT Link software during the instrument installation, your instrument data will be **automatically** imported into SMRT Link.

Several types of sequence data, as well as barcode files, can be imported for use in SMRT Link.

1. On the home page, select **Data Management**.
2. Click **Import Data**.
3. Specify whether to import data from the **SMRT Link Server**, or from a **Local File System**. (**Note: Only** references and barcodes are available if you select **Local File System**.)

Data Management / Import

Import Data

Import from

SMRT Link Server

Local File System

× Cancel

Select File

Data Type

- HiFi Reads or Subreads (XML)
- HiFi Reads or Subreads (ZIP)
- Barcodes (FASTA)
- Barcodes (XML)
- Barcodes (ZIP)
- References (FASTA)
- References (XML)
- References (ZIP)

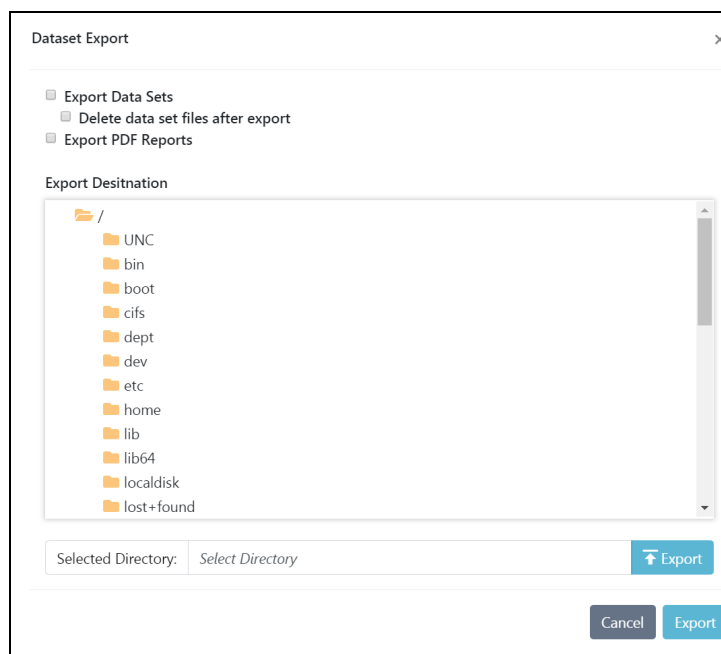
- Select the data type to import:
 - HiFi reads:** XML file (`.consensusreadset.xml`) or ZIP file containing information about HiFi reads (reads generated with CCS analysis whose quality value is equal to or greater than 20.) Use **only** ZIP files created by SMRT Link.
 - Subreads:** XML file (`.subreadset.xml`) or ZIP file containing information about subreads from Sequel II systems, such as paths to the BAM files. Use **only** ZIP files created by SMRT Link.
 - Barcodes:** FASTA (`.fa` or `.fasta`), XML (`.barcodeset.xml`), or ZIP files containing barcodes.
 - References:** FASTA (`.fa` or `.fasta`), XML (`.referenceSet.xml`), or ZIP files containing a reference sequence for use in starting analyses. (**Note:** If importing from a **local system**, Reference files must be smaller than 15 MB.)
 - Note:** FASTA files imported into SMRT Link must **not** contain empty lines or non-alphanumeric characters. The file name must **not** start with a number. For information about the file types listed here, click [here](#).
- Navigate to the appropriate file and click **Import**. The sequence data, reference, or barcodes are imported and becomes available in SMRT Link.

Exporting sequence, reference and barcode data

Two types of sequence data (HiFi reads and Subreads) can be exported, as well as barcode files and reference files.

- On the home page, select **Data Management**.
- Click **Export Data**.
- Select the type of data to export:
 - HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - Subreads:** Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

- **Barcodes:** Files containing barcodes.
 - **References:** Files containing a reference sequence for use in starting analyses.
4. **(Optional)** Use the Search function to search for Data Sets, barcode files, or reference files. See [“Appendix B - Data search” on page 147](#) for details.
 5. Select one or more sets of data to export. (Multiple data files are combined as one ZIP file for export.)
 6. Click **Export Selected**.



7. Navigate to the export destination directory.
8. **(Optional)** If exporting Data Sets, click **Delete data set files after export** to delete the Data Set(s) you selected from the SMRT Link installation. (Exporting, then deleting, Data Sets is useful for archiving Data Sets you no longer need.)
9. **(Optional)** If exporting Data Sets, click **Export PDF Reports** to create PDF files containing comprehensive information about the Data Set(s). Each PDF report contains extensive information about one Data Set, including loading statistics, run set up and QC information, analysis parameters and results including charts and histograms, and lists of the output files generated, all in one convenient document.
10. Click **Export**.

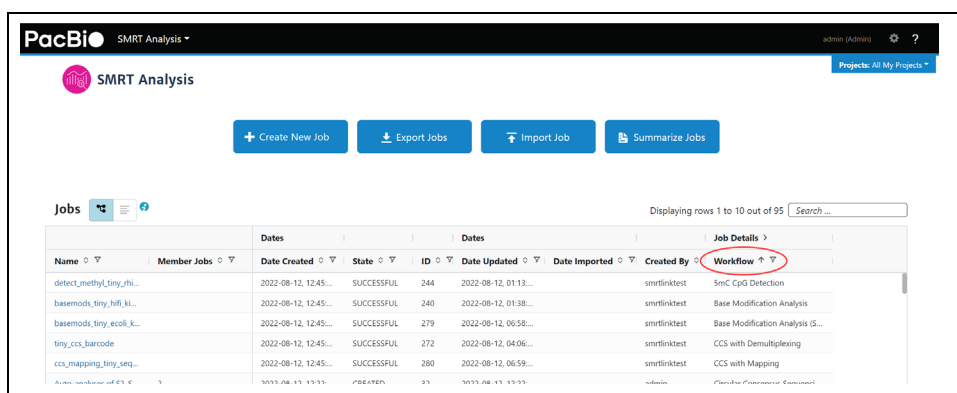
SMRT® Analysis

After a run has completed, use SMRT Link's **SMRT Analysis** module to perform **secondary analysis** of the data.

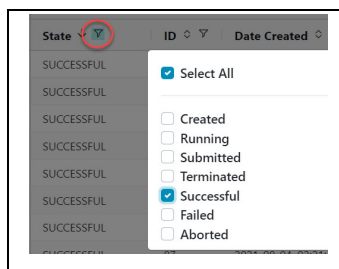
Creating and starting a job




1. Access SMRT Link using the Chrome web browser.
2. Select **SMRT Analysis**.



3. Jobs can be sorted, searched for, and filtered:
 - To **sort** jobs, click a **column title**.
 - To **search** for a job, use the Search function. See [“Appendix B - Data search” on page 147](#) for details.)
 - To **filter** the list of jobs based on their **state**: Click the funnel in the **State** column header, then click one or more of the categories of interest: **Select All**, **Created**, **Running**, **Submitted**, **Terminated**, **Successful**, **Failed**, or **Aborted**.



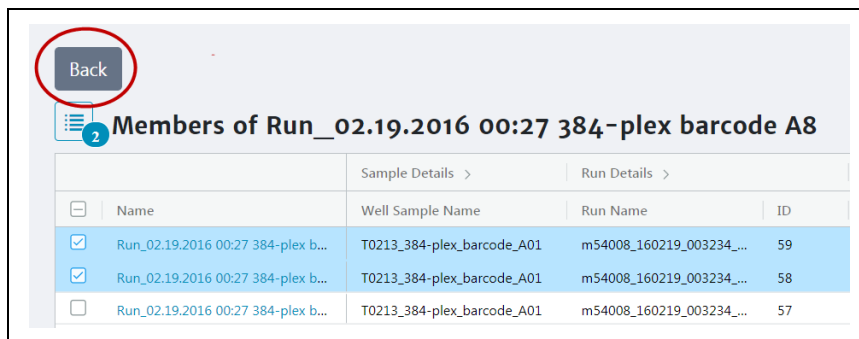
- To filter the list of jobs based on the **Project(s)** that they are associated with: Click the **Projects** menu (located at the top-right of the main SMRT Analysis page) and select a Project. See ["What is a Project?" on page 39](#) for details.
4. Click **+ Create New Job**.
 5. **(Optional)** Click **Copy From...**, choose a job whose settings you wish to reuse, then click **Select**. The job name and the Data Type are filled in. Go to Step 10 to select Data Set(s).
 6. Enter a **name** for the job.
 7. Specify the type of job to create:
 - **Analysis** - Uses applications designed to produce biologically-meaningful results. These applications **only** accept HiFi reads.
 - **Auto Analysis** - For information on the Auto Analysis feature, see ["Automated analysis" on page 126](#) for details.
 - **Data Utility** - Data processing utilities used as **intermediate steps** to producing biologically-meaningful results.
 8. If you selected **Data Utility**, select the type of data to use for the job:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.
 9. **(Optional)** Specify the **Project** that this job will be associated with using the **Projects** menu (located at the top-right of the SMRT Analysis page.) **General Project**: This job will be visible to **all** SMRT Link users. **All My Projects**: This job will be visible **only** to users who have access to Projects that you are a member of. To **restrict access** to a job, make sure to select a Project limited to the appropriate users **before** starting the job. **Note**: Selecting a Project **also** filters the Data Sets that you can use when **creating** the job.
 10. In the **Data Sets** table, select one or more sets of data to be analyzed.
 - **(Optional)** Use the Search function to search for Data Sets. See ["Appendix B - Data search" on page 147](#) for details.)
 - **(Optional)** Choose how to **view** the Data Set table: 1) Tree Mode - A barcoded Data Set displays as **one row**. 2) Flat Mode - A barcoded Data Set and its demultiplexed subsets display as **separate rows**.
 - **(Optional)** For Data Sets that include demultiplexed subsets, you can also select individual subsets as part of your selection. To do so:
 - A) Click the Demultiplexed Subsets number link:



Data Sets

<input type="checkbox"/>	Name	Demultiplexed Subsets
<input checked="" type="checkbox"/>	Re-barcode Alice/Bob/Charles	3
<input type="checkbox"/>	subreads-sequel	

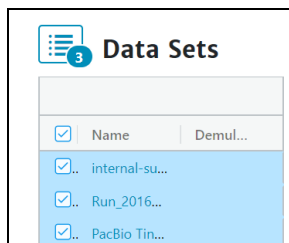
B) Select one or more subsets, then click **Back**:



Members of Run_02.19.2016 00:27 384-plex barcode A8

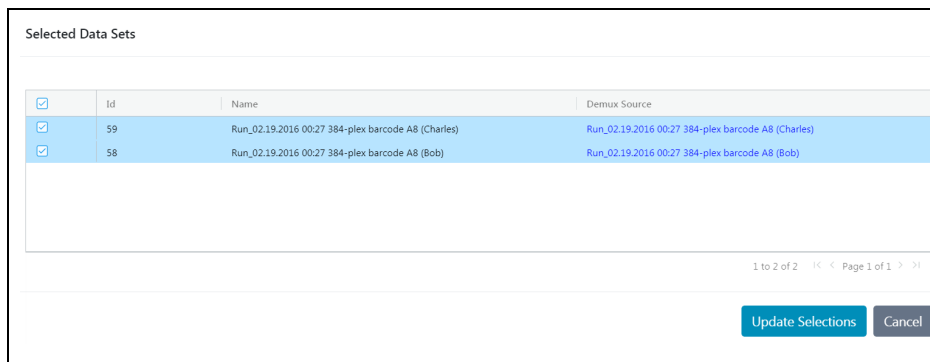
<input type="checkbox"/>	Name	Well Sample Name	Run Name	ID
<input checked="" type="checkbox"/>	Run_02.19.2016 00:27 384-plex b...	T0213_384-plex_barcode_A01	m54008_160219_003234_...	59
<input checked="" type="checkbox"/>	Run_02.19.2016 00:27 384-plex b...	T0213_384-plex_barcode_A01	m54008_160219_003234_...	58
<input type="checkbox"/>	Run_02.19.2016 00:27 384-plex b...	T0213_384-plex_barcode_A01	m54008_160219_003234_...	57

C) Click the list image to view or edit the full Data Set selection. (The small blue number specifies how many Data Sets and/or subsets were selected):



Data Sets

<input checked="" type="checkbox"/>	Name	Demul...
<input checked="" type="checkbox"/>	internal-su...	
<input checked="" type="checkbox"/>	Run_2016...	
<input checked="" type="checkbox"/>	PacBio Tin...	



Selected Data Sets

<input checked="" type="checkbox"/>	Id	Name	Demux Source
<input checked="" type="checkbox"/>	59	Run_02.19.2016 00:27 384-plex barcode A8 (Charles)	Run_02.19.2016 00:27 384-plex barcode A8 (Charles)
<input checked="" type="checkbox"/>	58	Run_02.19.2016 00:27 384-plex barcode A8 (Bob)	Run_02.19.2016 00:27 384-plex barcode A8 (Bob)

1 to 2 of 2 < > Page 1 of 1 >

Update Selections **Cancel**

Note: For information on the Auto Analysis feature, see [“Automated analysis” on page 126](#) for details.

11. If you selected **multiple** Data Sets as input for the job, additional options become available:

Workflow Type

☒ ANALYSIS ☐ AUTO ANALYSIS ☐ DATA UTILITY

Analysis of Multiple Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis for All Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis per Data Set - Custom Parameters

- **One Analysis for All Data Sets:** Runs one job using all the selected Data Sets as input, for a maximum of 30 Data Sets.
 - **One Analysis per Data Set - Identical Parameters:** Runs one separate job for **each** of the selected Data Sets, using the **same** parameters, for a maximum of 10,000 Data Sets. Later in the process, optionally click **Advanced Parameters** and modify parameters.
 - **One Analysis per Data Set - Custom Parameters:** Runs one separate job for **each** of the selected Data Sets, using **different** parameters for each Data Set, for a maximum of 16 Data Sets. Later in the process, click **Advanced Parameters** and modify parameters. Then click **Start and Create Next**. You can then specify parameters for **each** of the included Data Sets.
 - **Note:** The number of Data Sets listed is based on testing using PacBio's suggested compute configuration, listed in **SMRT Link software installation guide (v11.1)**.
12. Click **Next**.
13. Select a secondary analysis application or data utility from the drop-down list. (Different choices display based on your initial choice of **Analysis** or **Data Utility** in Step 7. See "[PacBio® secondary analysis applications](#)" on page 54 or "[PacBio® data utilities](#)" on page 99 for details.)

Analysis Application Required

Genome Assembly

HiFi Mapping

HiFiViral SARS-CoV-2 Analysis

Iso-Seq Analysis

Microbial Genome Analysis

Read Segmentation and Single-Cell Iso-Seq

Single-Cell Iso-Seq

Structural Variant Calling

- Each of the secondary analysis applications/data utilities has **required parameters** that are displayed. Review the default values shown.

- Secondary analysis applications/data utilities also have **advanced parameters**. These are set to default values, and need only be changed when analyzing data generated in non-standard experimental conditions.

14. **(Optional)** Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application or data utility. The imported settings are set.

The **Iso-Seq Analysis** application will be used as an example. This application characterizes full-length transcript isoforms.

15. Click the **Reference Set** field and select a reference sequence from the dialog. (The reference sequences available in SMRT Link and displayed in the dialog were imported into SMRT Analysis. See [“Importing sequence, reference and barcode data” on page 41](#) for details.)

SMRT Analysis / Create New Analysis

1. Select Data 2. Select Analysis

Analysis Application Required
Iso-Seq Analysis

Analysis Name
bbb

Import Analysis Settings Export

Associated Inputs

Primer Set Required
IsoSeqPrimers_v2

Reference Set

Run Clustering
☒ ON ☐ OFF

Cluster Barcoded Samples Separately
☐ ON ☒ OFF

Advanced Parameters

I..	Name
111	single-dataset-449

16. **(Optional)** Click **Advanced Parameters** and specify the values of the parameters you would like to change. Click **OK** when finished. (Different applications/data utilities have different advanced parameters.)

- To see information about parameters for **all** secondary analysis applications and data utilities provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#) and [“PacBio® data utilities” on page 99](#).

Advanced Analysis Parameters

Min. CCS Predicted Accuracy (Phred Scale) ⓘ <input type="text" value="20"/>	Require and Trim Poly(A) Tail ⓘ <input checked="" type="radio"/> ON <input type="radio"/> OFF	Minimum Mapped Length (bp) ⓘ <input type="text" value="50"/>
Minimum Gap-Compressed Identity (%) ⓘ <input type="text" value="95"/>	Minimum Mapped Coverage (%) ⓘ <input type="text" value="99"/>	Maximum Fuzzy Junction Difference (bp) ⓘ <input type="text" value="5"/>
Filters to Add to the Data Set ⓘ <input type="text"/>	Advanced pbmm2 Options ⓘ <input type="text"/>	Compute Settings ⓘ -- select --

Ok Cancel

17. **(Optional)** Click **Export** to create a CSV file containing **all** the settings you specified for the application/data utility. You can then import this file when creating future jobs using the same application/data utility. You can also use this exported file as a template for use with later jobs.
18. **(Optional)** Click **Back** if you need to change any of the analysis attributes selected in Step 7.
19. Click **Start** to submit the job. (If you selected multiple Data Sets as input, click **Start Multiple Jobs** or **Start and Create Next**.)
20. Select **SMRT Analysis** from the Module Menu to navigate to the main SMRT Analysis screen. There, the status of the job displays. When the job has **completed**, click on its name - reports are available for the completed job.
21. **(Optional)** To **delete** the completed job: Click **Delete**, then click **Yes** in the confirmation dialog. The job is deleted from **both** the SMRT Link interface and from the server.

SMRT Analysis

admin (Admin)

SMRT Analysis / Analysis Results

barcoding_tiny_sequel_manual

SUCCESSFUL

▼ Analysis Overview

Status

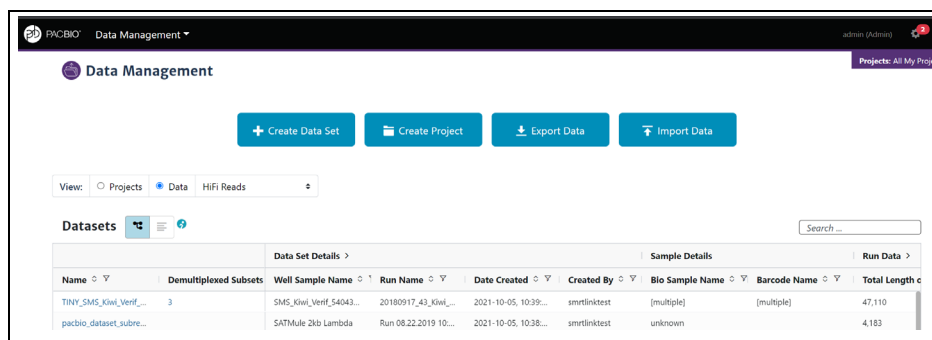
Analysis	Status
barcoding_tiny_sequel_manual	SUCCESSFUL: 6 tasks finished

Starting a job after viewing sequence data

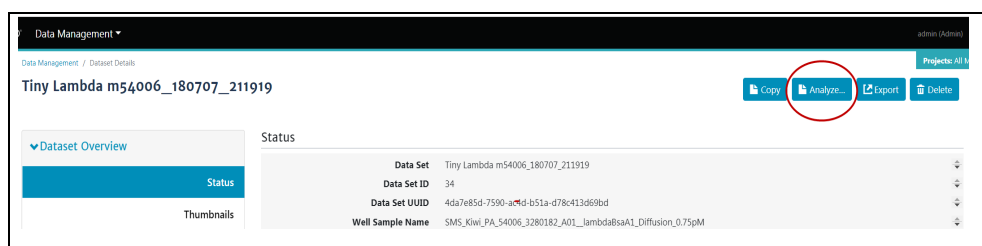
A job can be started by **first** viewing information about specific sequence data:

1. On the home page, select **Data Management**.
2. Click **View > Data** and select the type of Data Set to use:
 - **HiFi reads**: Reads generated with CCS analysis whose quality value is equal to or greater than 20.
 - **Subreads**: Reads containing the sequence from one or more single passes of a polymerase on a single strand of an insert within a SMRTbell template.

The Data Sets table displays the appropriate Data Sets available.
3. **(Optional)** Use the Search function to search for Data Sets. See [“Appendix B - Data search” on page 147](#) for details.



4. In the **Name** column, click the name of the sequence data of interest. Details for the selected sequence data display.



5. To **start** a job using this sequence data, click **Analyze...**, then name the job.
6. Select **Analysis** or **Utility** as the workflow type, then click **Next**.
7. Follow the instructions starting at Step 8 of ["Creating and starting a job"](#) on page 44.

Canceling a running job

1. On the home page, select **SMRT Analysis**.
2. Click the funnel in the **State** column header, then click **Running**. This displays **only** currently-running jobs.
3. Select a currently-running job to cancel.
4. Click **Cancel**.
5. Click **Yes** in the confirmation dialog. The canceled job displays as **Terminated**.

Restarting a failed job

You can **restart** a failed job; the execution speed from the start to the original point of failure is very fast, which can save time and computing resources. The restarted job **may** run to completion, depending on the source of failure.

Note: As the restarted job uses information from the original failed job, do **not** delete the original job results.

If viewing the results page for the failed job: Click **Restart**.

The screenshot shows the PacBio SMRT Analysis web interface. At the top, the user is logged in as 'mbudagyan (Admin)'. The main header shows 'Bsub_HGAP4 assembly' with a red 'FAILED' status and buttons for 'Send Log Files', 'Restart' (circled in red), 'Copy', and 'Delete'. On the left, a sidebar menu includes 'Analysis Overview', 'Status' (selected), 'Thumbnails', 'Display All', 'Coverage', 'Alignment to Draft Assembly', 'Preassembly', and 'Data'. The main content area displays job details for 'Bsub_HGAP4 assembly' (ID: 101448). The status is 'FAILED: Task genomic_consensus.tasks.variantcaller-10 FAILED in 1496.24 sec'. Other details include 'Created By: sdhillon', 'Date Created: 2019-09-04, 10:32:25 AM', 'Date Updated: 2019-09-04, 11:48:56 AM', 'Application: Assembly (HGAP 4)', 'SMRT Link Version: 7.1.0.71715', and 'Inputs: Sequel Data: Demux_32 plex_SD (bc1059--bc1059)'. The 'Path' is '/pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448'. The 'Error Message' states: 'Pbsmrtpipe job /pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448 failed with exit code 1. Task genomic_consensus.tasks.variantcaller-10 task genomic_consensus.tasks.variantcaller failed (exit-code 2) after 1496.24 sec Failed Task genomic_consensus.tasks.variantcaller exit code 2 in 20.37 sec (See file "/pbi/dept/secondary/siv/smartlink/smartlink-beta/smrtsuite_166987/userdata/jobs_root/101/101448/tasks/genomic_consensus.task:10/stderr").' It also notes 'Extracted from stderr: Unidentifiable sequencing chemistry present in dataset. Check if your SMRTanalysis installation is out-of-date. Unidentifiable sequencing chemistry present in dataset. Check if your'.

If **not** viewing the results page for the failed job:

1. On the home page, select **SMRT Analysis**.
2. Click the funnel in the **State** column header, then click **Failed**. This displays **only** failed jobs.
3. Select a failed job to restart.
4. Click **Restart**.

Viewing job results

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. (**Optional**) Click the funnel in the **State** column header, then click **Successful**. This displays **only** successfully-completed jobs.
3. (**Optional**) Use the Search function to search for specific jobs. See ["Appendix B - Data search" on page 147](#) for details.
4. Click the job link of interest.
5. Click **Analysis Overview > Status** to see job information status, including which application/data utility was used for the job, and the inputs used.
6. Click **Analysis Overview > Thumbnails** or **Display All** to view thumbnails of the reports generated for the job. Click the link under a thumbnail to see a larger image.
7. Depending on the application/data utility used for the job, different job-specific reports are available.
 - For mapping applications **only**: Click **Mapping Report > Summary Metrics** to see an overall summary of the mapping data.
 - For information on the reports and data files produced by analysis applications/data utilities, see ["PacBio® secondary analysis applications" on page 54](#) or ["PacBio® data utilities" on page 99](#).
8. To download data files created by SMRT Link: You can use these data files as input for further processing, pass on to collaborators, or upload to public genome sites. Click **Data > File Downloads**, then click

the appropriate file. The file is downloaded according to your browser settings.

9. **(Optional)** Specify prefix(es) used in the names of files generated by the job. Example: **Run Name** can be included in the name of every file generated by the job. Click **Edit Output File Name Prefix**, check the type(s) of information to append to the file names, then click **Save**.
10. To view job log details: Click **Data > SMRT Link Log**.
11. To visualize the secondary analysis results: See [“Visualizing data using IGV” on page 129](#) for details.

Copying and running an existing job

If you run very similar jobs, you can **copy** an existing job, rename it, optionally modify one or more parameters, then run it.

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. **(Optional)** Click the funnel in the **State** column header, then click **Successful**. This displays **only** successfully-completed jobs.
3. **(Optional)** Use the Search function to search for specific jobs. See [“Appendix B - Data search” on page 147](#) for details.
4. Click the job link of interest.
5. Click **Copy** - this creates a copy of the job, named `Copy of <job name>`, using the **same** parameters.
6. Edit the name of the job.
7. Click **Next**.
8. **(Optional)** Edit any other parameters. See [“PacBio® secondary analysis applications” on page 54](#) or [“PacBio® data utilities” on page 99](#) for further details.
9. Click **Start**.

Exporting a job

You can export the entire contents of a job directory, including the input sequence files, as a ZIP file. Afterwards, deleting the job saves room on the SMRT Link server; you can also later reimport the exported job into SMRT Link if necessary.

1. On the home page, select **SMRT Analysis**.
2. Click **Export Job**.
3. **(Optional)** Use the Search function to search for specific analyses. See [“Appendix B - Data search” on page 147](#) for details.
4. Select one or more jobs to export. This exports the entire contents of the job directory.
5. Click **Export Selected**.
6. Select the output directory for the job data and click **Export**.

Importing a job

Note: You can **only** import a job that was created in SMRT Link, then exported.

1. On the home page, select **SMRT Analysis**.

2. Click **Import Job**.
3. Select a ZIP file containing the job to import.
4. Click **Import**. The job is imported and is available on the main SMRT Analysis page.

Summarizing Microbial Genome Analysis jobs

You can summarize information about **multiple** successfully-completed Microbial Genome Analysis jobs. This generates the following:

- A CSV report with the assembly statistics for each job on a separate row.
 - A CSV report with the mapping statistics for each job on a separate row.
 - A CSV report with the coverage statistics for each contig for each job.
 - A text file containing a list of all the modified base motifs for each job.
 - A ZIP file containing the assembly in FASTA format from each job. (Available in the **Data > File Downloads** section of the results page.)
1. On the home page, select **SMRT Analysis**.
 2. Click **Summarize Jobs**. This displays all the successfully-completed Microbial Genome Analysis jobs.
 3. **(Optional)** Use the Search function to search for specific analyses. See ["Appendix B - Data search" on page 147](#) for details.
 4. Select one or more Microbial Genome Analysis jobs to summarize, up to 384 jobs.
 5. Click **Summarize Selected**. This creates a new job named **Job Summary**.
 6. When the **Job Summary** job is successfully completed, click it to see summary information about the Microbial Genome Analysis jobs you selected in Step 4.

PacBio® secondary analysis applications

Following are the secondary analysis applications provided with SMRT Analysis v11.1. These applications are designed to produce biologically-meaningful results. Each application is described later in this document, including all analysis parameters, reports and output files generated by the application.

Note: These applications accept **only** HiFi reads as input.

Genome Assembly

- Generate *de novo* assemblies of genomes, using HiFi reads.
- See [“Genome Assembly” on page 56](#) for details.

HiFi Mapping (was Mapping)

- Align (or map) reads to a user-provided reference sequence.
- See [“HiFi Mapping” on page 59](#) for details.

HiFiViral SARS-CoV-2 Analysis

- Analyze multiplexed viral surveillance samples for SARS-CoV-2, using HiFi reads.
- See [“HiFiViral SARS-CoV-2 Analysis” on page 63](#) for details.

Iso-Seq® Analysis

- Characterize full-length transcript isoforms, using HiFi reads.
- See [“Iso-Seq® Analysis” on page 68](#) for details.

Microbial Genome Analysis

- **Note:** This combines and replaces the **Microbial Assembly** and **Base Modification Analysis** applications in the previous release.
- Generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb, and identify methylated bases and associated nucleotide motifs.
- Optionally include identification of 6mA and 4mC modified bases and associated DNA sequence motifs.
- See [“Microbial Genome Analysis” on page 75](#) for details.

Read Segmentation and Single-Cell Iso-Seq® Analysis

- Characterize full-length transcript isoforms with additional single-cell information, including single-cell barcodes and unique molecular identifiers (UMIs).
- The application is for use when using concatenation-based library preparations such as the MAS-Seq libraries.
- See [“Read Segmentation and Single-Cell Iso-Seq® Analysis” on page 81](#) for details.

Single-Cell Iso-Seq® Analysis

- Characterize full-length transcript isoforms with additional single-cell information, including single-cell barcodes and unique molecular identifiers (UMIs).
- See for [“Single-Cell Iso-Seq® Analysis” on page 88](#) details.

Structural Variant Calling

- Identify structural variants (Default: ≥ 20 bp) in a sample or set of samples relative to a reference.
- See [“Structural Variant Calling” on page 95](#) for details.

Genome Assembly

Use this application to generate high quality *de novo* assemblies of genomes, using HiFi reads.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

The application includes seven main steps:

1. Convert input to a compressed database for fast retrieval.
2. Overlap reads using the **Pancake** tool.
3. Phase the overlapped reads using **Nighthawk**. Nighthawk also boosts contiguity of the assembly by removing overlaps between reads coming from different instances of a genomic repeat (such as segmental duplications.)
4. Remove chimeras and duplicate reads which do not span repeat regions. This improves contiguity and assembly quality.
5. Construct a string graph. Extract primary contigs and haplotigs. Haplotypes are represented by heterozygous bubbles.
6. Polish the contigs and haplotigs using phased reads. Phasing information is preserved. Polishing is done with [Racon](#).
7. Identify potential haplotype duplications in the primary contig set using the [purge_dups](#) tool, and move them to the haplotig set. This final round of assembly processing is especially useful in high heterozygosity samples.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Parameters

Advanced parameters	Default value	Description
Genome Length	0	The approximate number of base pairs expected in the genome. This is used only for downsampling; if the value is ≤ 0 , downsampling is disabled. Enter an integer, optionally followed by one of the metric suffixes: k, M or G. Example: 4500k means "4,500 kilobases" or "4,500,000". M stands for Mega and G stands for Giga.
Downsampled coverage	0	The input Data Set can be downsampled to a desired coverage, provided that both the Downsampled coverage and Genome Length parameters are specified and >0 . Downsampling applies to the entire assembly process, including polishing. This parameter selects reads randomly, using a fixed random seed for reproducibility.

Advanced parameters	Default value	Description
Run polishing	ON	Enables or disables the polishing stage of the workflow. Polishing can be disabled to perform fast draft assemblies.
Run phasing	ON	Enables or disables the phasing stage of the workflow. Phasing can be disabled to assemble haploid genomes, or to perform fast draft assemblies.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Advanced Assembly Options	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted.
Purge duplicate contigs from the assembly	ON	Enables or disables identification of "duplicate" alternate haplotype contigs which may be assembled in the primary contig file, and moves them to the associate contig (haplotig) file.
Cleanup intermediate files	ON	Removes intermediate files from the run directory to save space.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Genome Assembly application generates the following reports:

Polished Assembly > Summary Metrics

Displays statistics on the contigs from the *de novo* assembly that were corrected by Racon.

- **Contig Type:** Primary or Haplotigs. Primary contigs represent pseudohaplotype assemblies, while haplotigs represent fully phased and assembled regions of the genome. Primary contigs are usually much longer than haplotigs due to allowed haplotype switching.
- **Polished Contigs:** The number of polished contigs.
- **Maximum Contig Length:** The length of the longest contig.
- **Mean Contig Length:** The mean length of the contigs.
- **Median Contig Length:** The median length of the contigs.
- **N50 Contig Length:** 50% of the contigs are longer than this value.
- **Sum of Contig Lengths:** The total length of all the contigs.
- **E-size (sum of squares/sum):** The expected contig size for a random base in the polished contigs. Another interpretation: The area under the Nx curve (for x in range [0, 100]).
- **Number of Circular Contigs:** The number of assembled contigs that are circular.

Polished Assembly > Polished Contigs

- **Contig:** The name of the individual contig.
- **Length (bases):** The length of the contig, in bases.
- **Circular:** **Yes** if the contig is circular, **No** if it isn't.
- **Percent Polished:** The percent of contig bases that were polished.

- **Number of Polishing Reads:** The number of reads used to perform polishing on this contig.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Haplotigs:** The final polished haplotigs assembly, in FASTA format.
- **Primary Contigs:** The final polished primary contigs assembly, in FASTA format.

HiFi Mapping

Use this application to align (or map) data to a user-provided reference sequence. The HiFi Mapping application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Maps data to a provided reference sequence, and then identifies consensus and variants against this reference.
- Haploid variants and small indels, but **not** diploid variants, are called as a result to alignment to the reference sequence.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Set (Required)

- Specify a reference sequence to align the SMRT Cells reads to and to produce alignments.

Parameters

Advanced parameters	Default value	Description
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Minimum Mapped Length (bp)	50	The minimum required mapped read length, in base pairs.
Bio Sample Name of Aligned Dataset	NONE	Populates the Bio Sample Name (Read Group <i>SM</i> tag) in the aligned BAM file. If blank, uses the Bio Sample Name of the input file. Note: Avoid using spaces in Bio Sample Names as this may lead to third-party compatibility issues.
Minimum Gap-Compressed Identity (%)	70	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.1 for details.
Target Regions (BED file)	NONE	(Optional) Specifies a BED file that defines regions for a Target Regions report showing coverage over those regions. See “Appendix C - BED file format for Target Regions report” on page 149 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The HiFi Mapping application generates the following reports:

Target Regions > Target Regions

Displays the number (and percentage) of reads that hit target regions specified by an input BED file. This is useful for targeted DNA sequencing applications. (This report displays **only** if a BED file is specified when creating the analysis.)

- **Coordinates:** The chromosome coordinates, as specified in the input BED file.
- **Region:** The name of the region, as specified in the input BED file.
- **On-Target Reads:** The number (and percentage) of unique reads that map with any overlap to the target region.

Target Regions > Target Region Coverage

- Displays the number of hits per defined region of the chromosome.

Mapping Report > Summary Metrics

Mapping is local alignment of a read or subread to a reference sequence.

- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).
- **Number of Alignments:** The number of alignments that mapped to the reference sequence.
- **Number of CCS reads (total):** The total number of CCS reads in the sequence.
- **Number of CCS reads (mapped):** The number of CCS reads that mapped to the reference sequence.
- **Number of CCS reads (unmapped):** The number of CCS reads not mapped to the reference sequence.
- **Percentage of CCS reads (mapped):** The percentage of CCS reads that mapped to the reference sequence.
- **Percentage of CCS reads (unmapped):** The percentage of CCS reads not mapped to the reference sequence.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **CCS Read Length 95% (mapped):** The 95th percentile of read length of CCS reads that mapped to the reference sequence.
- **CCS Read Length Max (mapped):** The maximum length of CCS reads that mapped to the reference sequence.

Mapping Report > CCS Mapping Statistics Summary

Displays mapping statistics per movie.

- **Sample:** The sample name for which the following metrics apply.
- **Movie:** The movie name for which the following metrics apply.
- **Number of CCS Reads (mapped):** The number of CCS reads that mapped to the reference sequence. This includes adapters.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read Length N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped CCS Read Length

- Histogram distribution of the number of mapped CCS reads by read length.

Mapping Report > Mapped CCS Reads Concordance

- Histogram distribution of the number of CCS reads by the percent concordance with the reference sequence. Concordance for CCS reads is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped Concordance vs Read Length

- Maps the percent concordance with the reference sequence against the read length, in base pairs.

Coverage > Summary Metrics

- **Mean Coverage:** The mean depth of coverage across the reference sequence.
- **Missing Bases:** The percentage of the reference sequence without coverage.

Coverage > Coverage Across Reference

- Maps coverage across the reference.

Coverage > Depth of Coverage

- Maps the reference regions against the percent coverage.

Coverage > Coverage vs. [GC] Content

- Maps (as a percentage, over a 100 bp window) the number of Gs and Cs present across the coverage. The number of genomic windows with the corresponding % of Gs and Cs is displayed on top. Used to check that no coverage is lost over extremely biased base compositions.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Mapped Reads:** All input reads that were mapped to the reference by the application.
- **Coverage Summary:** Coverage summary for regions (bins) spanning the reference sequence.
- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see [“Visualizing data using IGV” on page 129](#) for details.

- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.

**HiFiViral
SARS-CoV-2
Analysis**

Use this application to analyze multiplexed samples sequenced with the HiFiViral SARS-CoV-2 kit. For **each** sample, this analysis provides:

- Consensus sequence (FASTA).
- Variant calls (VCF).
- HiFi reads aligned to the reference (BAM).
- Plot of HiFi read coverage depth across the SARS-CoV-2 genome.

Across **all** samples, this analysis provides:

- Job summary table including passing sample count at 90 and 95% genome coverage.
- Sample summary table including, for each sample: Count of variable sites, genome coverage, read coverage, and probability of multiple strains, and other metrics.
- Plate QC graphical summary of performance across samples in assay plate layout.
- Plot of HiFi read depth of coverage for all samples.

Notes:

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Phred-scaled Q20.
- This application is for SARS-CoV-2 analysis **only** and is **not** recommended for other viral studies. The Wuhan reference genome is provided by default to run the application, but advanced users may specify other reference genomes. We have **not** tested the application with reference genomes other than the Wuhan reference genome.
- The application is intended to identify variable sites and call a single consensus sequence per sample. The output consensus sequence is produced based on the dominant variant observed. Minor variant information that passes through a default threshold may be encoded in the raw VCF, but does **not** get propagated into the consensus sequence FASTA.
- The HiFiViral SARS-CoV-2 Analysis application can be run using the **Auto Analysis** feature available in Run Design. This feature allows users to complete all necessary analysis steps immediately after sequencing **without** manual intervention. The Auto Analysis workflow includes CCS, Demultiplex Barcodes, and HiFiViral SARS-CoV-2 Analysis.

Auto Analysis in Run Design

Users may set the analysis to begin **automatically** after sequencing completes using Auto Analysis in Run Design. See [“HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design” on page 127](#) for details.

HiFiViral SARS-CoV-2 application workflow

1. Process the reads using the `mimux` tool to trim the probe arm sequences.
2. Align the reads to the reference genome using `pbbmm2`.
3. Call and filter variants using `bcftools`, generating the raw variant calls in VCF file format. Filtering in this step removes low-quality calls (less than Q20), and normalizes indels.
4. Filter low-frequency variants using `vcfcons` and generate a consensus sequence by injecting variants into the reference genome. At each position, a variant is called **only** if **both** the base coverage **exceeds** the minimum base coverage threshold (Default = 4) **and** the fraction of reads that support this variant is **above** the minimum variant frequency threshold (Default = 0.5). See [here](#) for details.

Preparing input data for the HiFiViral SARS-CoV-2 Analysis application

1. Run the **Demultiplex Barcodes** data utility, where the inputs are HiFi reads, and the primers are multiplexed barcode primers. (If HiFi reads have **not** been generated on the instrument, run CCS analysis first. See [“Circular Consensus Sequencing \(CCS\)” on page 115](#) for details.)
 - The proper barcode sequences are provided by default:
`Barcoded M13 Primer Plate.`
 - For the **Same Barcodes on Both Ends of Sequence** parameter, specify **No**; the barcode pairs are **asymmetric**.
 - Provide the correctly-formatted barcode pair-to-Bio Sample CSV file for the **Assign Bio Sample Names to Barcodes** option. (For details, see [“Assign Bio Sample Names to Barcodes \(Required\)” on page 102.](#))

Running the HiFiViral SARS-CoV-2 Analysis application

1. **After** running the Demultiplex Barcode data utility, create a new job using **SMRT Analysis > Create New Job**.
2. Name the job.
3. Select all the demultiplex samples contained in the Data Set and choose **Analysis of Multiple Data Sets > One Analysis for All Data Sets**. Click **Next**.
4. Select **HiFiViral SARS-CoV-2 Analysis** from the Analysis Application list.
5. **SARS-CoV-2 Genome NC_045512.2** (the Wuhan reference genome) is automatically loaded; advanced users may select a different reference if desired.
6. To generate the optional Plate QC graphical summary, click **Advanced Parameters** and load a CSV file using the provided template (**assay-PlateQC_template_4by96.csv**) as a guide.
7. Click **OK**, then **Start**.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.

- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Genome (Required)

- Specify the full viral genome against which to align the reads and call variants. (The default is the Wuhan Reference genome.)

Parameters

Advanced parameters	Default value	Description
Plate QC CSV	NONE	(Optional) Specify a CSV file to generate the Plate QC report, which displays analysis results for each sample in the assay plate. The CSV file must contain barcode (asymmetric pairs), Bio Sample Name, assay plate IDs (can include 1-4 plates with unique names; avoid special characters), and assay plate well IDs in the format A01, A02, ...H12. (To create a new file, click Download Template , edit, and then save the CSV file.) The plate and well information corresponds to the location of samples during the SARS-CoV-2 enrichment assay.
Probes FASTA	NONE	Specify probe sequences in FASTA format if using probes other than the standard probes shipped in the HiFiViral SARS-CoV-2 Kit.
Minimum Base Coverage	4	Specify the minimum read depth at each position to report either a variant or a reference base. Positions with less than this specified coverage will have an N base output in the consensus sequence FASTA file. Increasing the minimum base coverage may result in more Ns and loss of variant detection. We do not recommend making this value lower than the default threshold of 4, as it may increase the number of false positive variants called.
Minimum Variant Frequency	0.5	Specify that only variants whose frequency is greater than this value are reported. This frequency is determined based on the read depth (DP) and allele read count (AD) information in the VCF output file. We recommend using the default value to properly call the dominant alternative variant while also filtering out potential artifacts.
Advanced Processing Options	NONE	Additional options to pass to the <code>mimux</code> preprocessing tool for trimming and filtering reads by probe sequences. Options should be entered in space-separated format. See the HiFiViral SARS-CoV-2 Analysis section of SMRT Tools reference guide (v11.1) for details.
Minimum Barcode Score	80	A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator of how well the chosen barcode pair matches. It ranges between 0 (no match) and 100 (a perfect match). This parameter specifies that reads with barcode scores below this minimum value are not included in analysis.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The HiFiViral SARS-CoV-2 Analysis application generates the following reports:

Summary Report > Summary Metrics

- **Samples:** The count of all input samples, whether or not they passed analysis.
- **Samples with Genome Coverage > 90%:** The number of samples where at least 90% of bases have at least four mapped reads overlapping their position.
- **Samples with Genome Coverage > 95%:** The number of samples where at least 95% of bases have at least four mapped reads overlapping their position.
- **Samples Failing Workflow:** The number of samples for which the analysis was unable to generate a per-sample report due to an absence of usable data.

Summary Report > Sample Summary

- **Bio Sample Name:** The name of the biological sample associated with the variants. (**Note:** Any spaces in the name are substituted by new line characters for consistency with output file names.)
- **Substitutions:** The count of all called substitutions in the consensus sequence for the sample.
- **Insertions:** The count of all called insertions in the consensus sequence for the sample.
- **Deletions:** The count of all called deletions in the consensus sequence for the sample.
- **Reads:** The total number of HiFi reads for the sample.
- **Read Coverage:** The mean number of mapped reads overlapping with each position in the reference genome.
- **On-Target Rate:** The mapping yield of reads; the number of unique mapped reads divided by the total number of reads.
- **Multiple Strains (Probability):** Samples are flagged as having multiple strains if the probability is at least 0.95. Samples may contain multiple strains due to sample contamination or presence of multiple strains in the RNA extract. To classify a sample as multi-strain, we tolerate error by using the binomial cumulative distribution function (with a fixed probability of 0.2). This feature is supported for samples with Ct < 26 with minor frequencies >20%. Samples **must** have >70% genome coverage to be called Multiple Strains.
- **Ns:** The number of bases in the consensus sequence that are Ns.
- **Genome Coverage:** The percentage of bases with at least four mapped reads overlapping their position by default. See the **Advanced Parameters** dialog to adjust minimum base coverage.

Summary Report > Genome Coverage

- Coverage plot showing the per-sample mean read coverage within a window of 100 bp. The shaded region displays the 25th to 75th percentile in the range of coverage across all samples, and the darker solid line displays the **median** coverage across all samples.

Summary Report > Plate QC

Plot showing analysis results for each plate cell used. This plot is generated **only** if the user supplies a Plate QC CSV file mapping Bio Sample Names to Well IDs in **Advanced Parameters**.

- **Blue** wells represent samples with at least 95% coverage.
- **Green** wells represent samples with at least 90% coverage.
- **Yellow** wells represent samples that passed the workflow but had genome coverage worse than 90%.
- **Red** wells represent samples that failed the workflow.

- **White** wells do **not** include a sample.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **All Samples, HiFi Reads FASTQ:** HiFi reads in FASTQ format for all samples.
- **All Samples, Consensus Sequence FASTA:** The full consensus genomic sequences; bases for which **no** consensus could be called are represented by **Ns**. See the **Advanced Parameters** dialog to adjust the minimum base coverage for outputting **Ns**.
- **All Samples, Genome Coverage Plots:** Plots for individual samples showing coverage depth across the genome.
- **All Samples, Variant Call VCF:** VCF file containing the final variant calls per sample.
- **All Samples, HiFi Reads Mapped BAM:** BAM file for each sample containing the HiFi reads aligned to the reference genome.
- **All Samples, Consensus Sequence Aligned BAM:** BAM file for each sample of consensus sequence aligned to the reference genome. The consensus sequence is split into fragments where there are **Ns** and each fragment is mapped.
- **All Samples, Raw Variant Calls VCF:** VCF file containing the intermediate variant calls per patient sample.
- **Sample Summary Table CSV:** CSV version of the data shown in the Sample Summary table.
- **All Samples, Probe Counts TSV:** Tab-delimited text file containing per-sample, per-probe counts. This file can be used to identify samples that are poorly sequenced or probes with high or low coverages.
- **Sample Inputs CSV:** CSV version of the Plate QC CSV, if supplied in the **Advanced Parameters** dialog.
- **Failed Sample Info:** CSV file containing information on samples for which the analysis was unable to generate a per-sample report due to an absence of usable data.
- **Failed Sample Info Analysis Logs:** Zipped log files describing failed samples.

**Iso-Seq®
Analysis**

The Iso-Seq application enables analysis and functional characterization of full-length transcript isoforms for sequencing data generated on PacBio instruments.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis that have a quality value equal to or greater than Q20.

Notes on Multiplexed Data

There are two ways in which an Iso-Seq library can be multiplexed:

1. Barcoded adapter Iso-Seq libraries

- If using the SMRTbell Barcoded Adapter with the Iso-Seq Express protocol **on or after** April 21, 2022, demultiplex the Data Set **prior** to running the Iso-Seq application.
- To analyze the samples in a single Iso-Seq run, select **all** the demultiplexed Data Sets to combine and begin the Iso-Seq analysis.
- If following the standard Iso-Seq Express protocol, select **Iso-Seq cDNA Primers** as the Primer Set.

2. Barcoded cDNA primer Iso-Seq libraries

- If following multiplexing guidelines using the Iso-Seq Express protocol **on or prior to** April 21, 2022 and you ordered synthesized oligos listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document **Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell prep kit 3.0**, demultiplex your Data Set using the Iso-Seq application. In other words, do **not** run the Demultiplexing Barcodes utility first.
- See the **Primer Set Selection** column below for the correct choice of primer sequences.

Multiplexed method	Demultiplexed before Iso-Seq?	Primer set selection
Not multiplexed	NO	Iso-Seq cDNA Primers
Barcoded adapters	YES	Iso-Seq cDNA Primers
Barcoded cDNA primer	NO	Iso-Seq 12 Barcoded cDNA Primers Or Custom cDNA Primers

The application includes three main steps:

1. **Classify:** Identify and remove primers (which may be cDNA primers or barcoded cDNA primers). Identify full-length reads based on the asymmetry of 5' and 3' primers. Trim off polyA tails and remove artifactual concatemers.
2. **Cluster (Optional):** Perform *de novo* clustering and consensus calling. Output full-length consensus isoforms that are further separated into high-quality (HQ) and low-quality (LQ) based on estimated accuracies.

3. **Collapse (Optional):** When a reference genome is selected, map HQ isoforms to the genome, then collapse redundant isoforms into unique isoforms.

To obtain full-length non-concatemer (FLNC) reads and **not** complete the Cluster step: Ensure that the **Run Clustering** option is set to **OFF**.

Iso-Seq determines two FLNC reads to be the same isoform, and will place them in the same cluster, if the two reads:

- Differ less than 100 bp on the 5' end.
- Differ less than 30 bp on the 3' end.
- Have no internal gaps that exceed 10 bp.

Iso-Seq will **only** output clusters that have at least two FLNC reads.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Primer Set (Required)

- Specify a primer sequence file in FASTA format to identify cDNA primers for removal. The primer sequence includes the 5' and 3' cDNA primers and (if applicable) barcodes.
- Primer IDs **must** be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should **not** include the Ts and is written in reverse complement (see examples below).
- Each primer sequence must be **unique**.

Example 1: The Iso-Seq cDNA Primer primer set, included with the SMRT Link installation.

Users following the standard Iso-Seq Express protocol **without** multiplexing, or running a Data Set that has **already** been demultiplexed (either using Run Design or the SMRT Analysis application) should use this default option.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>IsoSeq_3p
GTACTCTGCGTTGATACCACTGCTT
```

Example 2: The Iso-Seq 12 Barcoded cDNA Primers set, included with the SMRT Link installation.

Users using barcoded cDNA primers listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document **Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell prep kit 3.0**, should select this option.

```
>bc1001_5p
CACATATCAGAGTGC GGCAATGAAGTCGCAGGGTTGGG
>bc1002_5p
ACACACAGACTGTGAGGCAATGAAGTCGCAGGGTTGGG
...
```

(There are a total of 24 sequence records, representing 12 pairs of F/R barcoded cDNA primers.)

Example 3: An example of a custom cDNA primer set. 4 tissues were multiplexed using barcodes on the 3' end only.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>dT_BC1001_3p
AAGCAGTGGTATCAACGCAGAGTACCACATATCAGAGTGCG
>dT_BC1002_3p
AAGCAGTGGTATCAACGCAGAGTACACACACAGACTGTGAG
>dT_BC1003_3p
AAGCAGTGGTATCAACGCAGAGTACACACATCTCGTGAGAG
>dT_BC1004_3p
AAGCAGTGGTATCAACGCAGAGTACCACGCACACACGCGCG
```

Example 4: Special Handling for the TeloPrime cDNA Kit

The Lexogen TeloPrime cDNA kit contains **As** in the 3' primer that **cannot** be differentiated from the polyA tail. For best results, remove the **As** from the 3' end as shown below:

```
>TeloPrimeModified_5p
TGGATTGATATGTAATACGACTCACTATAG
>TeloPrimeModified_3p
CGCCTGAGA
```

Reference Set (Optional)

- Optionally specify a reference genome to align High Quality isoforms to, and to collapse isoforms mapped to the same genomic loci.

Run Clustering (Default = ON)

- Specify **ON** to generate consensus isoforms.
- Specify **OFF** to classify reads **only** and not generate consensus isoforms. The Reference Set will also be ignored.

Cluster Barcoded Samples Separately (Default = OFF)

- Specify **OFF** if barcoded samples are from the **same** species, but different tissues, or samples of the same genes but different individuals. The samples are clustered with **all** barcodes pooled.
- Specify **ON** if barcoded samples are from **different** species. The samples are clustered separately by barcode.
- In either case, the samples on the results page are automatically named `BioSample_1` through `BioSample_N`.

Parameters

Advanced parameters	Default value	Description
Require and Trim Poly(A) Tail	ON	ON means that polyA tails are required for a sequence to be considered full length. OFF means sequences do not need polyA tails to be considered full length.
Minimum Mapped Length (bp)	50	The minimum required mapped HQ isoform sequence length (in base pairs) for the Iso-Seq mapping-collapse step. Note: This is applicable only if a reference genome is provided.
Minimum Gap-Compressed Identity (%)	95	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference. Note: This is applicable only if a reference genome is provided.
Minimum Mapped Coverage (%)	99	The minimum required HQ transcript isoform sequence alignment coverage (in percent) for the Iso-Seq mapping-collapse step. Note: This is applicable only if a reference genome is provided.
Maximum Fuzzy Junction Difference (bp)	5	The maximum junction difference between two mapped isoforms to be collapsed into a single isoform. If the junction differences are all less than the provided value, they will all be collapsed. Setting to 0 requires all junctions to be exact to be collapsed into a single isoform. Applicable only if a reference genome is provided.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for Iso-Seq Analysis is 20 (QV 20), or 99% predicted accuracy.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. (pbmm2 is already running with <code>--preset ISOSEQ</code> .) Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.1 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Iso-Seq application generates the following reports:

CCS Analysis Read Classification > Summary Metrics

- **Reads:** The total number of CCS reads.
- **Reads with 5' and 3' Primers:** The number of CCS reads with 5' and 3' cDNA primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers:** The number of non-concatemer CCS reads with 5' and 3' primers detected.

- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail:** The number of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected. This is usually the number for full-length, non-concatemer (FLNC) reads, unless polyA tails are not present in the sample.
- **Mean Length of Full-Length Non-Concatemer Reads:** The mean length of the non-concatemer CCS reads with 5' and 3' primers and polyA tails detected.
- **Unique Primers:** The number of unique primers in the sequence.
- **Mean Reads per Primer:** The mean number of CCS reads per primer.
- **Max. Reads per Primer:** The maximum number of CCS reads per primer.
- **Min. Reads per Primer:** The minimum number of CCS reads per primer.
- **Reads without Primers:** The number of CCS reads without a primer.
- **Percent Bases in Reads with Primers:** The percentage of bases in CCS reads in the sequence data that contain primers.
- **Percent Reads with Primers:** The percentage of CCS reads in the sequence data that contain primers.

CCS Analysis Read Classification > Primer Data

- **Bio Sample Name:** The name of the biological sample associated with the primer.
- **Primer Name:** A string containing the pair of primer indices associated with this biological sample.
- **CCS Reads:** The number of CCS reads associated with the primer.
- **Mean Primer Quality:** The mean primer quality associated with the primer.
- **Reads with 5' and 3' Primers:** The number of CCS reads with 5' and 3' cDNA primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers:** The number of non-concatemer CCS reads with 5' and 3' primers detected.
- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail:** The number of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected. This is usually the number for full-length, non-concatemer (FLNC) reads, unless polyA tails are not present in the sample.

CCS Analysis Read Classification > Primer Read Statistics

- **Number Of Reads Per Primer:** Maps the number of reads per primer, sorted by primer ranking.
- **Primer Frequency Distribution:** Maps the number of samples with primers by the number of reads with primers.
- **Mean Read Length Distribution:** Maps the read mean length against the number of samples with primers.

CCS Analysis Read Classification > Primer Quality Scores

- Histogram of primer scores.

CCS Analysis Read Classification > Length of Full-Length Non-Concatemer Reads

- Histogram of the read length distribution of non-concatemer CCS reads with 5' and 3' primers and polyA tails detected.

Transcript Clustering > Summary Metrics

- **Sample Name:** The sample name for which the following metrics apply.
- **Number of High-Quality Isoforms:** The number of consensus isoforms that have an estimated accuracy **above** the specified threshold.
- **Number of Low-Quality Isoforms:** The number of consensus isoforms that have an estimated accuracy **below** the specified threshold.

Transcript Clustering > Length of Consensus Isoforms

- Histogram of the consensus isoform lengths and the distribution of isoforms exceeding a read length cutoff.

Transcript Mapping > Summary Metrics

- **Sample Name:** Sample name for which the following metrics apply.
- **Number of mapped unique isoforms:** The number of unique isoforms, where each unique isoform is generated by collapsing redundant HQ isoforms (such as those have very minor differences from one to one another) to one isoform. Each unique isoform may be generated from one or multiple HQ isoforms.
- **Number of mapped unique loci:** The number of unique mapped genomic loci among all unique isoforms. Multiple unique isoforms may map to the same genomic location, indicating these unique isoforms are transcribed from the same gene family, but spliced differently.

Transcript Mapping > Length of Mapped Isoforms

- Histogram of mapped isoforms binned by read length and the distribution of mapped isoforms exceeding a read length cutoff.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Primers Summary:** Text file listing how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- **Inferred Primers:** Inferred primers used in the analysis. The algorithm looks at the first 35,000 ZMWs, then selects primers with ≥ 10 counts and mean scores ≥ 45 .
- **Full-Length Non-Concatemer Reads:** Full-length reads that have primers and polyA tails removed, in BAM format.
- **Full-Length Non-Concatemer Report:** Includes strand, 5' primer length, 3' primer length, polyA tail length, insertion length, and primer IDs for each full-length read that has primers and polyA tail, in CSV format.
- **Low-Quality Isoforms:** Isoforms with low consensus accuracy, in FASTQ and FASTA format. We recommend that you work only with High-Quality isoforms, unless there are specific reasons to analyze Low-Quality isoforms. When the input Data Set is a ConsensusReadSet, a FASTA file **only** is generated.
- **High-Quality Isoforms:** Isoforms with high consensus accuracy, in FASTQ and FASTA format. This is the recommended output file to work with. When the input Data Set is a ConsensusReadSet, a FASTA file **only** is generated.
- **Cluster Report:** Report of each full-length read into isoform clusters.
- **Isoform Counts by Barcode:** For each isoform, report supportive FLNC reads for each barcode.
- **Mapped High Quality Isoforms:** Alignments mapping isoforms to the reference genome, in BAM and BAI (index) formats.
- **Collapsed Filtered Isoforms GFF:** Mapped, unique isoforms, in GFF format. This is the Mapping step output that is the recommended output file to work with.

- **Collapsed Filtered Isoforms:** Mapped, unique isoforms, in FASTQ format. This is the Mapping step output that is recommended output file to work with. When the input Data Set is a ConsensusReadSet, **only** a FASTA file is generated.
- **Collapsed Filtered Isoforms Groups:** Report of isoforms mapped into collapsed filtered isoforms.
- **Full-length Non-Concatemer Read Assignments:** Report of full-length read association with collapsed filtered isoforms, in text format.
- **Collapsed Filtered Isoform Counts:** Report of read count information for each collapsed filtered isoform.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see [“Visualizing data using IGV” on page 129](#) for details.

- **Mapped High Quality Isoforms:** Alignments mapping isoforms to the reference genome, in BAM and BAI (index) formats.

Note: For details on custom PacBio tags added to output BAM files by the Iso-Seq Application, see page 54 of **SMRT Tools reference guide (v11.1)**, or see [here](#) for details.

**Microbial
Genome
Analysis**

Use this application to generate *de novo* assemblies of small prokaryotic genomes between 1.9-10 Mb and companion plasmids between 2 – 220 kb. This application can optionally include analysis of 6mA and 4mC modified bases and associated DNA sequence motifs. (This requires kinetic information.)

Note: This combines and replaces the **Microbial Assembly** and **Base Modification Analysis** applications in the previous release.

The Microbial Genome Analysis application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Includes chromosomal- and plasmid-level *de novo* genome assembly, circularization, polishing, and rotation of the origin of replication for each circular contig.
- Performs base modification detection to identify 4mCm and 6mA and associated DNA sequence motifs. (This requires kinetic information.)
- Facilitates assembly of larger genomes (yeast) as well.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Run Base Modification Analysis (Default = ON)

- Run Base Modification analysis on the final assembly. This **only** applies if the assembly is not empty, and the input data contains the correct kinetic tags.

Find Modified Base Motifs (Default = ON)

- Perform motif detection on the results of base modification analysis.

Parameters

Advanced parameters	Default value	Description
Downsampled coverage	0	The input Data Set can be downsampled to a desired coverage, provided that the Downsampled coverage value is >0. Downsampling applies to the entire assembly process, including polishing. This parameter selects reads randomly, using a fixed random seed for reproducibility.
Advanced Assembly Options for chromosomal stage	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. See Appendix C in SMRT Tools reference guide (v11.1) for details.
Advanced Assembly Options for plasmid stage	NONE	A semicolon-separated list of KEY=VALUE pairs. New line characters are not accepted. See Appendix C in SMRT Tools reference guide (v11.1) for details.
Maximum plasmid length, bp	300,000	Value that should be set higher than the maximum size of a plasmid in the input sample. The default value should work well in most cases.
Run secondary polish	ON	Specify that an additional polishing stage be run at the end of the workflow.
Base modifications to identify	m4C,m6A	Specify the base modifications to identify, in a comma-separated list.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Cleanup intermediate files	ON	Removes intermediate files from the run directory to save space.
Minimum Qmod Score	35	Specify the minimum Qmod score to use in motif-finding.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Microbial Genome Analysis application generates the following reports:

Mapping Report > Summary Metrics

Mapping is local alignment of a read or subread to a reference sequence.

- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).
- **Number of Alignments:** The number of alignments that mapped to the reference sequence.
- **Number of CCS reads (total):** The total number of CCS reads in the sequence.
- **Number of CCS reads (mapped):** The number of CCS reads that mapped to the reference sequence.
- **Number of CCS reads (unmapped):** The number of CCS reads not mapped to the reference sequence.

- **Percentage of CCS reads (mapped):** The percentage of CCS reads that mapped to the reference sequence.
- **Percentage of CCS reads (unmapped):** The percentage of CCS reads not mapped to the reference sequence.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read Length N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **CCS Read Length 95% (mapped):** The 95th percentile of read length of CCS reads that mapped to the reference sequence.
- **CCS Read Length Max (mapped):** The maximum length of CCS reads that mapped to the reference sequence.

Mapping Report > CCS Mapping Statistics Summary

Displays mapping statistics per movie.

- **Sample:** The sample name for which the following metrics apply.
- **Movie:** The movie name for which the following metrics apply.
- **Number of CCS Reads (mapped):** The number of CCS reads that mapped to the reference sequence. This includes adapters.
- **CCS Read Length Mean (mapped):** The mean read length of CCS reads that mapped to the reference sequence, starting from the first mapped base of the first mapped CCS read, and ending at the last mapped base of the last mapped CCS read.
- **CCS Read Length N50 (mapped):** The read length at which 50% of the mapped bases are in CCS reads longer than, or equal to, this value.
- **Number of CCS Bases (mapped):** The number of CCS bases that mapped to the reference sequence.
- **Mean Concordance (mapped):** The mean concordance of subreads that mapped to the reference sequence. Concordance for alignment is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped CCS Read Length

- Histogram distribution of the number of mapped CCS reads by read length.

Mapping Report > Mapped CCS Reads Concordance

- Histogram distribution of the number of CCS reads by the percent concordance with the reference sequence. Concordance for CCS reads is defined as the number of matching bases over the number of alignment columns (match columns + mismatch columns + insertion columns + deletion columns).

Mapping Report > Mapped Concordance vs Read Length

- Maps the percent concordance with the reference sequence against the CCS read length, in base pairs.

Polished Assembly > Summary Metrics

Displays statistics on the contigs from the *de novo* assembly that were corrected by Arrow.

- **Polished Contigs:** The number of polished contigs.
- **Maximum Contig Length:** The length of the longest contig.
- **N50 Contig Length:** 50% of the contigs are longer than this value.
- **Sum of Contig Lengths:** Total length of all the contigs.
- **E-size (sum of squares/sum):** The expected contig size for a random base in the polished contigs.

Polished Assembly > Polished Contigs from Microbial Assembly HiFi

Displays a table of details about all assembled contigs.

- **Contig:** The contig name.
- **Length:** The length of the contig, in base pairs, after polishing.
- **Circular:** Marks whether circularity of the contig was detected. Output values are `yes` and `no`.
- **Coverage:** The average coverage across the contig, calculated by the sum of coverage of all bases in the contig divided by the number of bases.

Coverage > Summary Metrics

Displays depth of coverage across the *de novo*-assembled genome, as well as depth of coverage distribution.

- **Mean Coverage:** The mean depth of coverage across the assembled genome sequence.
- **Missing Bases:** The percentage of the genome's sequence that have zero depth of coverage.

Coverage > Coverage across Reference

- Displays coverage at each position of the draft genome assembly.

Coverage > Depth of Coverage

- Histogram distribution of the draft assembly regions by the coverage.

Coverage > Coverage vs. [GC] Content

- Maps (as a percentage, over a 100 bp window) the number of Gs and Cs present across the coverage. The number of genomic windows with the corresponding % of Gs and Cs is displayed on top. Used to check that no coverage is lost over extremely biased base compositions.

Base Modifications > Kinetic Detections

- **Per-Base Kinetic Detections:** Maps the modification QV against per-strand coverage.
- **Kinetic Detections Histogram:** Histogram distribution of the number of bases by modification QV.

Modified Base Motifs > Modified Base Motifs

Displays statistics for the methyltransferase recognition motifs detected.

- **Motif:** The nucleotide sequence of the methyltransferase recognition motif, using the standard IUPAC nucleotide alphabet.
- **Modified Position:** The position within the motif that is modified. The first base is 0. **Example:** The modified adenine in GATC is at position 2.
- **Modification Type:** The type of chemical modification most commonly identified at that motif. These are: 6mA, 4mC, or `modified_base` (modification not recognized by the software.)
- **% of Motifs Detected:** The percentage of times that this motif was detected as modified across the entire genome.
- **# of Motifs Detected:** The number of times that this motif was detected as modified across the entire genome.
- **# of Motifs In Genome:** The number of times this motif occurs in the genome.
- **Mean QV:** The mean modification QV for all instances where this motif was detected as modified.
- **Mean Coverage:** The mean coverage for all instances where this motif was detected as modified.
- **Partner Motif:** For motifs that are not self-palindromic, this is the complementary sequence.
- **Mean IPD Ratio:** The mean inter-pulse duration. An IPD ratio greater than 1 means that the sequencing polymerase slowed down at this base position, relative to the control. An IPD ratio less than 1 indicates speeding up.
- **Group Tag:** The motif group of which the motif is a member. Motifs are grouped if they are mutually or self reverse-complementary. If the motif isn't complementary to itself or another motif, the motif is given its own group.
- **Objective Score:** For a given motif, the objective score is defined as $(\text{fraction methylated}) * (\text{sum of log-p values of matches})$.

Modified Base Motifs > Modification QVs

- Maps motif sites against Modification QV for all genomic occurrences of a motif, for each reported motif, including "No Motif".

Modified Base Motifs > ModQV Versus Coverage by Motif

- Maps coverage against Modification QV for all genomic occurrences of a motif, for each reported motif.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Per-Base Kinetics:** CSV file containing per-base information.
- **Per-Base IPDs for IGV:** BigWig file containing encoded per-base IPD ratios.
- **Motif Annotations:** GFF file listing every modified nucleotide sequence motif in the genome.
- **Modified Base Motifs:** CSV file containing statistics for the methyltransferase recognition motifs detected.
- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.
- **Modified Bases:** GFF file listing every detected modified base in the genome.

- **Final Polished Assembly:** The polished assembly before oriC rotation is applied, in FASTA format.
- **Final Polished Assembly Index:** The BAI index file for the polished assembly before oriC rotation is applied.
- **Final Polished Assembly for NCBI:** The final polished assembly with applied oriC rotation and header adjustment for NCBI submission, in FASTA format.
- **Coverage Summary:** Coverage summary for regions (bins) spanning the reference sequence.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see [“Visualizing data using IGV” on page 129](#) for details.

- **Mapped BAM:** The BAM file of subread alignments to the draft contigs used for polishing.
- **Mapped BAM Index:** The BAI index file for the corresponding Mapped BAM file.
- **Final Polished Assembly:** The polished assembly before oriC rotation is applied, in FASTA format.
- **Final Polished Assembly Index:** The BAI index file for the polished assembly before oriC rotation is applied.
- **Per-Base IPDs for IGV:** BigWig file containing encoded per-base IPD ratios.

Read Segmentation and Single-Cell Iso-Seq® Analysis

The Read Segmentation and Single-Cell Iso-Seq Analysis application enables analysis and functional characterization of full-length transcript isoforms with additional single-cell information, including single-cell barcodes and unique molecular identifiers (UMIs), that were sequenced on PacBio instruments.

The application is for use when using concatenation-based library preparations such as the MAS-Seq libraries.

The Read Segmentation and Single-Cell Iso-Seq Analysis application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Workflow

1. Split arrayed HiFi reads at adapter positions, generating **segmented reads** (S-reads) which are the comprising fragments. For each input HiFi read, the step creates multiple BAM records, one for each fragment. An arrayed HiFi read can contain many fragments.
2. Full-length reads are then identified by the presence of cDNA primers and polyA tails. Then, UMI and barcode information is extracted.
3. After barcode correction and UMI deduplication, the unique molecules are mapped to the reference genome and classified and filtered against reference annotation using `pigeon`, which is a transcript classification and filtering tool based on the **SQANTI3** software.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Segmentation Adapter Set

- Specify a FASTA file, provided by PacBio, containing segmentation adapters. If you need a **custom** segmentation adapter set, click **Advanced Parameters** and use a custom FASTA file formatted as described in the table below.

Primer Set (Required) (Default = 10x Chromium single cell 3' cDNA primers)

- Specify a primer sequence file in FASTA format to identify cDNA primers for removal. The primer sequence includes the 5' and 3' cDNA primers.

- Primer IDs **must** be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should **not** include the Ts and is written in reverse complement. (See the example below.)
- Each primer sequence must be **unique**.

Example: The 10x Chromium single cell 3' cDNA primer set.

```
>5p
AAGCAGTGGTATCAACGCAGAGTACATGGG
>3p
AGATCGGAAGAGCGTCGTGTAG
```

Reference Set (Required)

- Specify one of two default reference genome and annotation sets to align high quality isoforms to, and to collapse isoforms mapped to the same genomic loci. The default sets are `Human_hg38_Gencode_v39` and `Mouse_mm39_Gencode_vM28`.

Parameters

Advanced parameters	Default value	Description
10X Barcodes (text, gzipped)	3m-february-2018.txt.gz	A gzipped text file containing known 10X single-cell barcodes, one per line. This include list specifies the barcode-set to which raw cell barcodes are remapped by minimum edit distance.
Adapters FASTA	NONE	Specify a custom FASTA file containing segmentation adapters. If not specified, the adapters specified in the XML metadata are used. Adapters must be ordered in the expected order of adapters in the reads. There should be one entry per adapter (forward or reverse-complement orientation) with no overlapping adapter sequences. Duplicate names or sequences are not allowed. Example: >A AGCTTACTTGTGAAGA >B ACTTGTAAGCTGTCTA >C ACTCTGTCAGGTCCGA >D ACCTCCTCCTCCAGAA >E AACCGGACACACTTAG
Single cell barcode and UMI design	T-12U-16B	<ul style="list-style-type: none"> • <code>T</code> indicates the transcript position and is mandatory. It is the anchor that determines whether tags are located on the 3' or 5' side. • <code>U</code>, as in UMI, must be preceded by the length of the UMI. • <code>B</code>, as in cell barcode, must be preceded by the length of the cell barcode. Example: <code>T-12U-16B</code> indicates a 12bp UMI and 16bp cell barcode after the transcript (with polyA tail).

Advanced parameters	Default value	Description
Output prefix	isoseq	The output file name prefix for all non-matrix files. If not specified, the primary input classification file is used to determine the prefix. Example: <code>my_isoforms_classification.txt</code> inputs yields an output prefix of <code>my_isoforms</code> .
Cell Barcode Finding Method	knee	Select the <code>knee</code> or <code>percentile</code> method to determine the number of real cells based on the cell barcode ranking plot.
Cell Barcode Percentile Cutoff	99	An integer between 0-100 for the percentile cutoff if using the <code>percentile</code> cell barcode finding method. (This value is ignored if using the <code>knee</code> cell barcode finding method.)
Base task memory (MB)	512	Specifies the minimum memory per task when submitting a job.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Read Segmentation and Single-Cell Iso-Seq Analysis application generates the following reports:

Read Segmentation > Summary Metrics

- **Reads:** The number of input arrayed HiFi reads.
- **Segmented reads (S-reads):** The number of generated S-reads.
- **Mean length of S-reads:** The mean read length of the generated S-reads.
- **Percent of reads with full arrays:** The percentage of input reads containing all adapter sequences in the order listed in the segmentation adapter FASTA file.
- **Mean array size:** The mean number of fragments (or S-reads) found in the input reads.

Read Segmentation > Segmentation Statistics

- Histogram distribution of the number of S-reads per read.
- Heatmap of adapter ligations.

Read Segmentation > Length of Reads

- Histogram distribution of the number of HiFi reads by read length, in base pairs.

Read Segmentation > S-read Length Distribution

- Histogram distribution of the number of S-reads by the HiFi read length, in base pairs.

Read Statistics > Summary Metrics

- **Reads:** The total number of input reads.
- **Read Type:** The type of input reads - `CCS`, `SEGMENT`, or `mixed` if there are multiple input data sets with mixed data types.
- **Reads with 5' and 3' Primers with extracted UMIs and Barcodes:** The number of reads with 5' and 3' cDNA primers detected, and UMI/cell barcode information extracted. Also known as full-length tagged reads (FLT Reads).
- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail (FLNC Reads):** The number of non-concatemer reads with 5' and 3' primers and polyA tails detected after UMI/cell barcode information has been extracted.

- **FLNC Reads with Valid Barcodes:** The number of full-length non-concatemer tagged reads that include valid barcodes, given a cell barcode whitelist.
- **FLNC Reads with Valid Barcodes, corrected:** The number of full-length non-concatemer tagged reads that include valid barcodes (given a cell barcode whitelist) after cell barcode correction.
- **Reads after Barcode Correction and UMI Deduplication:** The number of deduplicated reads, after barcode correction and deduplication.

Read Statistics > Length of Reads

- Histogram distribution of the number of input reads by read length.

Cell Statistics > Summary Metrics

- **Estimated Number of Cells:** The estimated number of cells.
- **Reads in Cells:** The percentage of reads in cells.
- **Mean Reads per Cell:** The mean number of reads per cell.
- **Median UMIs per Cell:** The median number of unique molecular identifiers (UMIs) per cell.

Cell Statistics > Barcode Rank Plot

- Displays the distribution of barcode counts and which barcodes were inferred to be associated with cells. The x-axis denotes barcodes ranked in decreasing order by UMI counts mapped to each barcode, and the y-axis denotes the UMI count for the x-th ranked barcode.

Transcript Statistics > Summary Metrics

- **FLNC reads mapped confidently to genome:** The number of FLNC reads mapped to the reference genome. This number is calculated first based on the number of deduplicated reads mapped to the genome, then expanded to account for duplicate FLNC reads for each unique molecule.
- **FLNC reads mapped confidently to transcriptome:** The number of FLNC reads mapped to the reference genome in which the read is later associated with a transcript that is classified as one of the following: FSM, ISM, NIC, or NNC.
- **Total unique genes:** The total number of unique genes across all cells.
- **Total unique genes, filtered:** The total number of unique genes, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique genes, known genes only:** The total number of unique genes across all cells in which the gene is annotated in the reference annotation.
- **Total unique genes, filtered, known genes only:** The total number of unique genes (genes annotated in the reference annotation) across all cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts:** The total number of unique transcripts across all cells.
- **Total unique transcripts, filtered:** The total number of unique transcripts across all cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all cells in which the gene the transcript belongs to is annotated in the reference annotation.
- **Total unique transcripts, filtered, known transcripts only:** The total number of unique transcripts across all cells, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes (genes annotated in the reference annotation) are included.

Transcript Statistics > Transcript Summary

- **Median genes per cell:** The median number of genes per cell.

- **Median genes per Cell, known genes only:** The median number of unique, known genes (genes annotated in the reference annotation) per input cell.
- **Median transcripts per cell:** The median number of transcripts per cell.
- **Median transcripts per cell, known transcripts only:** The median number of transcripts per cell. Only transcripts associated with known genes are included.
- **Total unique genes:** The total number of unique genes across all cells.
- **Total unique genes, known genes only:** The total number of unique, known genes (genes annotated in the reference annotation) across all cells.
- **Total unique transcripts:** The total number of unique transcripts across all cells.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all cells. Only transcripts associated with known genes are included.

Transcript Statistics > Transcript Summary, Filtered

- **Median genes per cell:** The median number of genes per cell, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Median genes per cell, known genes only:** The median number of unique known genes (genes annotated in the reference annotation) per cell, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Median transcripts per cell:** The median number of transcripts per cell, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Median transcripts per cell, known transcripts only:** The median number of transcripts per cell, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes are included.
- **Total unique genes:** The total number of unique genes across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique genes, known genes only:** The total number of unique known genes (genes annotated in the reference annotation) across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts:** The total number of unique transcripts across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes are included.

Transcript Statistics > Transcript Classification

- **Category:** Transcript classification assigned by the classification and filtering tool `pigeon`, based on the [SQANTI3](#) software.

Category	Description
FSM (Full splice match)	The reference and query isoform have the same number of exons and each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
ISM (Incomplete splice match)	The query isoform has fewer external exons than the reference, but each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
NIC (Novel in catalog)	The query isoform does not have a FSM or ISM match, but is using a combination of known donor/acceptor sites.
NNC (Novel not in catalog)	The query isoform does not have a FSM or ISM match, and has at least one donor or acceptor site that is not annotated.
Antisense	The query isoform does not have overlap a same-strand reference gene but is anti-sense to an annotated gene.

Category	Description
Fusion	The query isoform overlaps two or more reference genes.
More junctions	The query isoform overlaps two or more reference genes, however some junctions are shared by multiple genes.
Genic intron	The query isoform is completely contained within a reference intron.
Genic genomic	The query isoform overlaps with introns and exons.
Intergenic	The query isoform is in the intergenic region.

- **Count:** The number of transcripts in a specific classification.
- **CAGE Detected:** The number of transcripts where the transcription start site falls within 50bp of an annotated CAGE (Cap Analysis of Gene Expression) peak site. (See [here](#) for more information.)
- **CAGE Detected (%):** The percentage of transcripts where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **polyA Motif Detected:** The number of transcripts where a known polyA motif is detected upstream of the transcription end site.
- **polyA Motif Detected (%):** The percentage of transcripts where a known polyA motif is detected upstream of the transcription end site.

Transcript Statistics > Transcript Classification, filtered

- **Category:** Transcript classification assigned by the classification and filtering tool *pigeon*, based on the [SQANTI3](#) software.
- **Count:** The number of transcripts, after filtering out reads based on the SQANTI filtering criteria, in a specific classification.
- **CAGE Detected:** The number of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **CAGE Detected (%):** The percentage of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **polyA Motif Detected:** The number of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where a known polyA motif is detected upstream of the transcription end site.
- **polyA Motif Detected (%):** The percentage of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where a known polyA motif is detected upstream of the transcription end site.

Transcript Statistics > Transcript Classification Plots

- **Isoform distributions across structural categories:**
 - Distribution of the percentage of transcripts by structural categories.
- **Structural categories by isoform lengths:**
 - Length distribution of transcripts in different structural categories.

Transcript Statistics > Transcript Classification Plots, Filtered

- **Isoform distributions across structural categories:**
 - Distribution of the percentage of isoforms by structural categories, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Structural categories by isoform lengths:**
 - Histogram display of the number of isoforms by their length in KB and their structural category, after filtering out reads based on the SQANTI transcript filtering criteria.

Transcript Statistics > Gene Saturation

Saturation plot showing the level of gene saturation based on the number of subsampled reads.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log.**)
- **Report read_segmentation:** JSON report containing summary statistics.
- **Segmented Reads, passing, unaligned:** BAM file containing the generated S-reads that passed filtering.
- **Non-passing reads, unaligned:** BAM file containing HiFi reads that did **not** generate S-reads.
- **Transcript Classifications:** Text file containing detailed information on transcripts.
- **Transcript Exon Junctions:** Text file containing detailed information on the exon junctions of each transcript.
- **Mapped Transcripts BAM:** BAM file containing the transcripts that mapped to the reference genome.
- **Mapped Transcripts BAM Index:** BAM index file associated with the Mapped Transcript BAM file.
- **Cell barcode ranking and count information, after correction:** Tab-delimited CSV file containing single-cell barcode ranking and count information after cell barcode correction.
- **Unique mapped transcripts, GFF:** GFF file containing unique mapped transcripts.
- **Unique mapped transcripts, filtered, GFF:** GFF file containing unique mapped transcripts after filtering.
- **Unique mapped transcripts, classification TXT:** Text file containing unique mapped transcript classifications against annotations.
- **Unique mapped transcripts, filtered, classification TXT:** Text file containing unique mapped transcript classifications against annotations, after filtering.
- **Unique mapped transcripts, junctions TXT:** Text file containing information about unique mapped transcript junctions.
- **Unique mapped transcripts, filtered, junctions TXT:** Text file containing information about unique mapped transcript junctions, after filtering.
- **Deduplicated reads after cell barcode correction, unmapped, BAM:** BAM file containing unmapped reads after cell barcode correction and UMI deduplication.
- **Deduplicated reads after cell barcode correction, mapped, BAM:** BAM file containing mapped reads after cell barcode correction and UMI deduplication.
- **Deduplicated reads after cell barcode correction, mapped, BAM index:** BAM index file associated with the BAM file containing mapped reads after cell barcode correction and UMI deduplication.
- **Single-cell isoform and gene matrix, gzipped:** Gzipped file containing Seurat-compatible isoform and gene matrix files.
- **<Data Set> Segmented Reads:** Output Data Set, containing generated S-reads and supplementary files.

Single-Cell Iso-Seq® Analysis

The Single-Cell Iso-Seq application enables analysis and functional characterization of full-length transcript isoforms with additional single cell information, including single cell barcodes and UMIs, that were sequenced on PacBio instruments.

The Single-Cell Iso-Seq Analysis application:

- Accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Primer Set (Required) (Default = 10x Chromium single cell 3' cDNA primers)

- Specify a primer sequence file in FASTA format to identify cDNA primers for removal. The primer sequence includes the 5' and 3' cDNA primers and (if applicable) barcodes.
- Primer IDs **must** be specified using the suffix `_5p` to indicate 5' cDNA primers and the suffix `_3p` to indicate 3' cDNA primers. The 3' cDNA primer should **not** include the Ts and is written in reverse complement (see examples below).
- Each primer sequence must be **unique**.

Example 1: The Iso-Seq cDNA Primer primer set, included with the SMRT Link installation.

Users following the standard Iso-Seq Express protocol **without** multiplexing, or running a Data Set that has **already** been demultiplexed (either using Run Design or the SMRT Analysis application) should use this default option.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>IsoSeq_3p
GTACTCTGCGTTGATACCACTGCTT
```

Example 2: The Iso-Seq 12 Barcoded cDNA Primers set, included with the SMRT Link installation.

Users using barcoded cDNA primers listed in the **Appendix 3 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section of the document

Procedure & checklist - Preparing Iso-Seq® libraries using SMRTbell prep kit 3.0, should select this option.

```
>bc1001_5p
CACATATCAGAGTGC GGCAATGAAGTCGCAGGGTTGGGG
>bc1002_5p
ACACACAGACTGTGAGGCAATGAAGTCGCAGGGTTGGGG
...
```

(There are a total of 24 sequence records, representing 12 pairs of F/R barcoded cDNA primers.)

Example 3: An example of a custom cDNA primer set. 4 tissues were multiplexed using barcodes on the 3' end only.

```
>IsoSeq_5p
GCAATGAAGTCGCAGGGTTGGG
>dT_BC1001_3p
AAGCAGTGGTATCAACGCAGAGTACCACATATCAGAGTGC
>dT_BC1002_3p
AAGCAGTGGTATCAACGCAGAGTACACACACAGACTGTGAG
>dT_BC1003_3p
AAGCAGTGGTATCAACGCAGAGTACACACATCTCGTGAGAG
>dT_BC1004_3p
AAGCAGTGGTATCAACGCAGAGTACCACGCACACACGCGCG
```

Example 4: Special Handling for the TeloPrime cDNA Kit

The Lexogen TeloPrime cDNA kit contains **As** in the 3' primer that **cannot** be differentiated from the polyA tail. For best results, remove the **As** from the 3' end as shown below:

```
>TeloPrimeModified_5p
TGGATTGATATGTAATACGACTCACTATAG
>TeloPrimeModified_3p
CGCCTGAGA
```

Reference Set (Required)

- Specify one of two default reference genome sets to align High Quality isoforms to, and to collapse isoforms mapped to the same genomic loci. The default sets are `Human_hg38_Gencode_v39` and `Mouse_mm39_Gencode_vM28`.

Parameters

Advanced parameters	Default value	Description
10X Barcodes (text, gzipped)	3m-february-2018.txt.gz	Gzipped text file containing the allowed barcodes, one barcode per line.
Output prefix	isoseq	The output file name prefix for all non-matrix files. If not specified, the primary input classification file is used to determine the prefix. Example: <code>my_isoforms_classification.txt</code> inputs yields an output prefix of <code>my_isoforms</code> .

Advanced parameters	Default value	Description
Single cell barcode and UMI design	T-12U-16B	<ul style="list-style-type: none"> T indicates the transcript position and is mandatory. It is the anchor that determines whether tags are located on the 3' or 5' side. U, as in UMI, must be preceded by the length of the UMI. B, as in cell barcode, must be preceded by the length of the cell barcode. <p>Example: T-12U-16B indicates a 12bp UMI and 16bp cell barcode after the transcript (with polyA tail).</p>
Cell Barcode Finding Method	knee	Select the <i>knee</i> or <i>percentile</i> method to determine the number of real cells based on the cell barcode ranking plot.
Cell Barcode Percentile Cutoff	99	An integer between 0-100 for the percentile cutoff if using the <i>percentile</i> cell barcode finding method. (This value is ignored if using the <i>knee</i> cell barcode finding method.)
Base task memory (MB)	512	Specifies the minimum memory per task when submitting a job.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Single-Cell Iso-Seq Analysis application generates the following reports:

Read Statistics > Summary Metrics

- **Reads:** The total number of input reads.
- **Read Type:** The type of input reads - *CCS*, *SEGMENT*, or *mixed* if there are multiple input data sets with mixed data types.
- **Reads with 5' and 3' Primers with extracted UMIs and Barcodes:** The number of reads with 5' and 3' cDNA primers detected, and UMI/cell barcode information extracted. Also known as full-length tagged reads (FLT Reads).
- **Non-Concatemer Reads with 5' and 3' Primers and Poly-A Tail (FLNC Reads):** The number of non-concatemer reads with 5' and 3' primers and polyA tails detected after UMI/cell barcode information has been extracted.
- **FLNC Reads with Valid Barcodes:** The number of full-length non-concatemer tagged reads that include valid barcodes, given a cell barcode whitelist.
- **FLNC Reads with Valid Barcodes, corrected:** The number of full-length non-concatemer tagged reads that include valid barcodes (given a cell barcode whitelist) after cell barcode correction.
- **Reads after Barcode Correction and UMI Deduplication:** The number of deduplicated reads, after barcode correction and deduplication.

Read Statistics > Length of Reads

- Histogram distribution of the number of HiFi reads by read length.

Cell Statistics > Summary Metrics

- **Estimated Number of Cells:** The estimated number of real cells, based on the cell barcode ranking plot.
- **Reads in Cells:** The number of reads in cells.
- **Mean Reads per Cell:** The mean number of reads per cell.
- **Median UMIs per Cell:** The median number of unique molecular identifiers (UMIs) per cell.

Cell Statistics > Barcode Rank Plot

- Displays the distribution of barcode counts and which barcodes were inferred to be associated with cells. The x-axis denotes barcodes ranked in decreasing order by UMI counts mapped to each barcode, and the y-axis denotes the UMI count for the x-th ranked barcode.

Transcript Statistics > Summary Metrics

- **FLNC reads mapped confidently to genome:** The number of FLNC reads mapped to the reference genome. This number is calculated first based on the number of deduplicated reads mapped to the genome, then expanded to account for duplicate FLNC reads for each unique molecule.
- **FLNC reads mapped confidently to transcriptome:** The number of FLNC reads mapped to the reference genome in which the read is later associated with a transcript that is classified as one of the following: FSM, ISM, NIC, or NNC.
- **Total unique genes:** The total number of unique genes across all cells.
- **Total unique genes, filtered:** The total number of unique genes, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique genes, known genes only:** The total number of unique genes across all cells in which the gene is annotated in the reference annotation.
- **Total unique genes, filtered, known genes only:** The total number of unique genes (genes annotated in the reference annotation) across all cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts:** The total number of unique transcripts across all cells.
- **Total unique transcripts, filtered:** The total number of unique transcripts across all cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all cells in which the gene the transcript belongs to is annotated in the reference annotation.
- **Total unique transcripts, filtered, known transcripts only:** The total number of unique transcripts across all cells, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes (genes annotated in the reference annotation) are included.

Transcript Statistics > Transcript Summary

- **Median genes per cell:** The median number of genes per cell.
- **Median genes per cell, known genes only:** The median number of unique, known genes (genes annotated in the reference annotation) per input cell.
- **Median transcripts per cell:** The median number of transcripts per cell.
- **Median transcripts per cell, known transcripts only:** The median number of transcripts per cell. Only transcripts associated with known genes are included.
- **Total unique genes:** The total number of unique genes across all cells.
- **Total unique genes, known genes only:** The total number of unique, known genes (genes annotated in the reference annotation) across all cells.
- **Total unique transcripts:** The total number of unique transcripts across all cells.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all cells. Only transcripts associated with known genes are included.

Transcript Statistics > Transcript Summary, Filtered

- **Median genes per cell:** The median number of genes per cell, after filtering out reads based on the SQANTI transcript filtering criteria.

- **Median genes per cell, known genes only:** The median number of unique known genes (genes annotated in the reference annotation) per cell, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Median transcripts per cell:** The median number of transcripts per cell, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Median transcripts per cell, known transcripts only:** The median number of transcripts per cell, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes are included.
- **Total unique genes:** The total number of unique genes across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique genes, known genes only:** The total number of unique known genes (genes annotated in the reference annotation) across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts:** The total number of unique transcripts across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Total unique transcripts, known transcripts only:** The total number of unique transcripts across all input cells, after filtering out reads based on the SQANTI transcript filtering criteria. Only transcripts associated with known genes are included.

Transcript Statistics > Transcript Classification

- **Category:** The type of transcript detected.

Category	Description
FSM (Full splice match)	The reference and query isoform have the same number of exons and each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
ISM (Incomplete splice match)	The query isoform has fewer external exons than the reference, but each internal junction matches the positions of the reference. The exact 5' start and 3' end can differ within the first/last exons.
NIC (Novel in catalog)	The query isoform does not have a FSM or ISM match, but is using a combination of known donor/acceptor sites.
NNC (Novel not in catalog)	The query isoform does not have a FSM or ISM match, and has at least one donor or acceptor site that is not annotated.
Antisense	The query isoform does not have overlap a same-strand reference gene but is anti-sense to an annotated gene.
Fusion	The query isoform overlaps two or more reference genes.
More junctions	The query isoform overlaps two or more reference genes, however some junctions are shared by multiple genes.
Genic intron	The query isoform is completely contained within a reference intron.
Genic genomic	The query isoform overlaps with introns and exons.
Intergenic	The query isoform is in the intergenic region.

- **Count:** The number of transcripts in a specific classification.
- **CAGE Detected:** The number of transcripts where the transcription start site falls within 50bp of an annotated CAGE (Cap Analysis of Gene Expression) peak site. (See [here](#) for more information.)
- **CAGE Detected (%):** The percentage of transcripts where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **polyA Motif Detected:** The number of transcripts where a known polyA motif is detected upstream of the transcription end site.
- **polyA Motif Detected (%):** The percentage of transcripts where a known polyA motif is detected upstream of the transcription end site.

Transcript Statistics > Transcript Classification, filtered

- **Category:** Transcript classification assigned by the classification and filtering tool `pigeon`, based on the `SQANTI3` software.
- **Count:** The number of transcripts, after filtering out reads based on the SQANTI filtering criteria, in a specific classification.
- **CAGE Detected:** The number of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **CAGE Detected (%):** The percentage of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where the transcription start site falls within 50bp of an annotated CAGE peak site.
- **polyA Motif Detected:** The number of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where a known polyA motif is detected upstream of the transcription end site.
- **polyA Motif Detected (%):** The percentage of transcripts, after filtering out reads based on the SQANTI transcript filtering criteria, where a known polyA motif is detected upstream of the transcription end site.

Transcript Statistics > Transcript Classification Plots

- **Isoform distributions across structural categories:**
 - Distribution of the percentage of isoforms by structural categories.
- **Structural categories by isoform lengths:**
 - Histogram display of the number of isoforms by their length in KB and their structural category.

Transcript Statistics > Transcript Classification Plots, Filtered

- **Isoform distributions across structural categories:**
 - Distribution of the percentage of isoforms by structural categories, after filtering out reads based on the SQANTI transcript filtering criteria.
- **Structural categories by isoform lengths:**
 - Histogram display of the number of isoforms by their length in KB and their structural category, after filtering out reads based on the SQANTI transcript filtering criteria.

Transcript Statistics > Gene Saturation

Saturation plot showing the level of gene saturation based on the number of subsampled reads.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Transcript Classifications:** Text file containing detailed information on transcripts.
- **Transcript Exon Junctions:** Text file containing detailed information on the exon junctions of each transcript.
- **Mapped Transcripts BAM:** BAM file containing the transcripts that mapped to the reference genome.
- **Mapped Transcripts BAM Index:** BAM index file associated with the Mapped Transcript BAM file.

- **Cell barcode ranking and count information, after correction:** Tab-delimited CSV file containing single-cell barcode ranking and count information after cell barcode correction.
- **Unique mapped transcripts, GFF:** GFF file containing unique mapped transcripts.
- **Unique mapped transcripts, filtered, GFF:** GFF file containing unique mapped transcripts after filtering.
- **Unique mapped transcripts, classification TXT:** Text file containing unique mapped transcript classifications against annotations.
- **Unique mapped transcripts, filtered, classification TXT:** Text file containing unique mapped transcript classifications against annotations, after filtering.
- **Unique mapped transcripts, junctions TXT:** Text file containing information about unique mapped transcript junctions.
- **Unique mapped transcripts, filtered, junctions TXT:** Text file containing information about unique mapped transcript junctions, after filtering.
- **Deduplicated reads after cell barcode correction, unmapped, BAM:** BAM file containing unmapped reads after cell barcode correction and UMI deduplication.
- **Deduplicated reads after cell barcode correction, mapped, BAM:** BAM file containing mapped reads after cell barcode correction and UMI deduplication.
- **Deduplicated reads after cell barcode correction, mapped, BAM index:** BAM index file associated with the BAM file containing mapped reads after cell barcode correction and UMI deduplication.
- **Single-cell isoform and gene matrix, gzipped:** Gzipped file containing Seurat-compatible isoform and gene matrix files.

Structural Variant Calling

Use this application to identify structural variants (Default: ≥ 20 bp) in a sample or set of samples relative to a reference. Variant types identified are insertions, deletions, duplications, copy number variants (CNVs), inversions, and translocations.

- The application accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected application. The imported application settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the application. You can then import this file when creating future analyses using the same application. You can also use this exported file as a template for use with later analyses.

Reference Set (Required)

- Specify a reference genome against which to align the reads and call variants.

Parameters

Advanced parameters	Default value	Description
Minimum Length of Structural Variant (bp) (Required)	20	The minimum length of structural variants, in base pairs.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Minimum % of Reads that Support Variant (any one sample) (Required)	10	Ignore calls supported by <P% of reads in every sample.
Minimum Reads that Support Variant (any one sample) (Required)	3	Ignore calls supported by <N reads in every sample.
Minimum Reads that Support Variant (total over all samples) (Required)	3	Ignore calls supported by <N reads total across samples.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Minimum Mapped Length (bp)	50	The minimum required mapped read length, in base pairs.
Minimum Gap-Compressed Identity (%)	70	The minimum required gap-compressed alignment identity, in percent. Gap-compressed identity counts consecutive insertion or deletion gaps as one difference.
Bio Sample Name of Aligned Dataset	NONE	Populates the Bio Sample Name (Read Group SM tag) in the aligned BAM file. If blank, uses the Bio Sample Name of the input file. Note: Avoid using spaces in Bio Sample Names as this may lead to third-party compatibility issues.

Advanced parameters	Default value	Description
Advanced pbmm2 Options	NONE	Space-separated list of custom pbmm2 options. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.1 for details.
Advanced pbsv Options	NONE	Additional pbsv command-line arguments. See SMRT® Tools reference guide v11.1 for details.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

To launch a multi-sample analysis

1. Click + **Create New Job**.
2. Enter a **name** for the analysis.
3. Ensure that **Analysis** is selected as the workflow type.
4. Select all the Data Sets for all the input samples.
5. In the **Analysis of Multiple Data Sets** list, select **One Analysis for All Data Sets**.
6. Click **Next**.
7. Select **Structural Variant Calling** from the Analysis Application list.

Note: The Data Set field **Bio Sample Name** identifies which Data Sets belong to which biological samples.

- If **multiple** Data Sets with the same Bio Sample Name are selected and submitted, the Structural Variant Calling application **merges** those Data Sets as belonging to the same sample.
- If any input Data Sets do **not** have a Bio Sample Name specified, they are merged (if there are multiple such Data Sets) and their Bio Sample Name is set to `UnnamedSample` in the analysis results.

Reports and data files

The Structural Variant Calling application generates the following reports:

Report > Count by Sample (SV Type)

This table describes the type of called variants broken down by individual sample. For each sample, only variants for which the sample has a heterozygous ("0/1") or homozygous alternative ("1/1") genotype are considered.

- **Insertions (total bp):** The count and total length (in base pairs) of all called insertions in the sample.
- **Deletions (total bp):** The count and total length (in base pairs) of all called deletions in the sample.
- **Inversions (total bp):** The count and total length (in base pairs) of all called inversions in the sample.
- **Translocations:** The count of all called translocations in the sample.
- **Duplications (total bp):** The count and total length (in base pairs) of all called duplications in the sample.
- **Total Variants (total bp):** The count and total length (in base pairs) of all variants in the sample.

Report > Count by Sample (Genotype)

This table describes the genotype of called variants broken down by individual sample. For each sample, only variants for which the sample has a heterozygous ("0/1") or homozygous alternative ("1/1") genotype are considered.

- **Homozygous Variants:** The count of homozygous variants called in the sample.
- **Heterozygous Variants:** The count of heterozygous variants called in the sample.
- **Total Variants:** The count of all called variants in the sample.

Report > Count by Annotation

This table describes the called variants broken down by a set of repeat annotations. Each variant is counted once (regardless of sample genotypes) and assigned to exactly **one** annotation category. Only insertion and deletion variants are considered in this report.

- **Tandem repeat:** Variant sequence is a short pattern repeated directly next to itself.
- **ALU:** Variant sequence matches the ALU SINE repeat consensus.
- **L1:** Variant sequence matches the L1 LINE repeat consensus.
- **SVA:** Variant sequence matches the SVA LINE repeat consensus.
- **Unannotated:** Variant sequence does **not** match any of the above patterns.
- **Total:** The sum of variants from all annotations.

Report > Length Histogram

- Histogram of the distribution of variant lengths, in base pairs, broken down by individual. For each individual, separate distributions are provided for variants between 10-99 base pairs, 100-999 base pairs, and ≥ 1 kilobase pairs. Each variant is counted once, regardless of sample genotypes.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Aligned Reads (per sample):** Aligned reads, in BAM format, separated by individual.
- **Index of Aligned Reads (per sample):** BAM index files associated with the Aligned Reads BAM files.
- **Structural Variants:** All the structural variants, in VCF format.

Data > IGV Visualization Files

The following files are used for visualization using IGV; see ["Visualizing data using IGV" on page 129](#) for details.

- **Aligned Reads (per sample):** Aligned reads, in BAM format, separated by individual.
- **Index of Aligned Reads (per sample):** BAM index files associated with the Aligned Reads BAM files.
- **Structural Variants:** All the structural variants, in VCF format. (See [here](#) for details.)

PacBio® data utilities

Following are data processing utilities provided with SMRT Analysis v11.1. These utilities are used as intermediate steps to producing biologically-meaningful results. Each utility is described later in this document, including all parameters, reports and output files generated by the utility.

Note: The following data utilities accept **only** HiFi reads as input.

5mC CpG Detection

- Analyze the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.
- See [“5mC CpG Detection” on page 100](#) for details.

Demultiplex Barcodes

- Separate reads by barcode.
- See [“Demultiplex Barcodes” on page 101](#) for details.

Export Reads

- Export HiFi reads that pass filtering criteria as FASTA, FASTQ and BAM files.
- For **barcoded** runs, you must **first** run the **Demultiplex Barcodes** application to create BAM files **before** using this application.
- See [“Export Reads” on page 107](#) for details.

Mark PCR Duplicates

- Remove duplicate reads from a HiFi reads Data Set created using an ultra-low DNA sequencing protocol.
- See [“Mark PCR Duplicates” on page 109](#) for details.

Read Segmentation

- Splits arrayed HiFi reads at adapter positions, generating segmented reads (S-reads) comprised of multiple fragments.
- See [“Read Segmentation” on page 111](#) for details.

Trim Ultra-Low Adapters (was Trim gDNA Amplification Adapters)

- Trim PCR Adapters from a HiFi reads Data Set created using an ultra-low DNA sequencing library.
- See [“Trim Ultra-Low Adapters” on page 113](#) for details.

Note: The following data utility accept **only** Subreads as input.

Circular Consensus Sequencing (CCS)

- Identify consensus sequences for single molecules.
- See [“Circular Consensus Sequencing \(CCS\)” on page 115](#) for details.

5mC CpG Detection

Use this utility to analyze the kinetic signatures of cytosine bases in CpG motifs to identify the presence of 5mC.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20. The utility also requires kinetics information.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Parameters

Advanced parameters	Default value	Description
Keep Kinetics in Output	OFF	If ON, specifies that the IPD and PulseWidth records are included in the output BAM file.
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The 5mC CpG Detection utility generates the following reports:

5mC CpG Report > Methylation Probability

- **CpG Methylation in Reads:** The cumulative of percentage of CpG sites in the sample mapped against the predicted probability of methylation.
- **CpG Methylation in Reads (Histogram):** Histogram displaying the percentage of CpG sites in the sample versus the predicted probability of methylation.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **HiFi reads with 5mC Calls:** BAM file containing all the HiFi reads in the sample that include 5mC calls.
- **<Input Data Set>(5mC):** Output Data Set with the 5mC calls.

Demultiplex Barcodes

Use this utility to separate sequence reads by barcode. (See [“Working with barcoded data” on page 118](#) for more details.)

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- Barcoded SMRTbell templates are SMRTbell templates with adapters flanked by barcode sequences, located on both ends of an insert.
- For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For **asymmetric** designs, **different** barcodes are attached to the sides of the insert sequence of interest.
- Barcode names and sequences, independent of orientation, **must** be unique.
- Most-likely barcode sequences per SMRTbell template are identified using a FASTA-format file of the known barcode sequences.

Given an input set of barcodes and a BAM Data Set, the Demultiplex Barcodes utility produces:

- A set of BAM files whose reads are annotated with the barcodes;
- A `ConsensusReadSet` file that contains the file paths of that collection of barcode-tagged BAM files and their related files.

Notes on Iso-Seq Multiplexed Data

There are two ways in which an Iso-Seq library can be multiplexed:

1. Barcoded adapter Iso-Seq libraries

- If using the SMRTbell Barcoded Adapter with the Iso-Seq Express protocol **on or after** April 21, 2022, demultiplex the Data Set **prior** to running the Iso-Seq application.
- To analyze the samples in a **single** Iso-Seq run, select **all** the demultiplexed Data Sets to combine and begin the Iso-Seq analysis.

2. Barcoded cDNA primer Iso-Seq libraries

- If following multiplexing guidelines using the Iso-Seq Express protocol **prior to** April 21, 2022 and you ordered synthesized oligos listed in the **Appendix 2 - Recommended barcoded NEBNext single cell cDNA PCR primer and Iso-Seq Express cDNA PCR primer sequences** section, demultiplex your Data Set using the Iso-Seq application. In other words, do **not** run the Demultiplexing Barcodes utility first.
- See [“Iso-Seq® Analysis” on page 68](#) for choices on Primer Sets to use.

Multiplexed method	Run Demultiplex Barcodes utility?
Not multiplexed	NO
Barcoded adapters	YES
Barcoded cDNA primer	NO

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Barcode Set (Required)

- Specify a barcode sequence file to separate the reads.

Same Barcodes on Both Ends of Sequence (Default = Yes)

- Specify **Yes** to retain all the reads with the **same** barcodes on both ends of the insert sequence, such as symmetric and tailed designs. (See [“Working with barcoded data” on page 118](#) for information on barcode designs.)
- Specify **No** to specify asymmetric designs where the barcodes are **different** on each end of the insert sequence.

Assign Bio Sample Names to Barcodes (Required)

SMRT Link automatically creates a CSV-format **Autofilled Barcoded Sample** file. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names 1 through N , corresponding to barcodes 1 through N , for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names to barcodes:

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the checkboxes to select multiple barcodes.)
- (Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid spaces in Bio Sample Names as they may lead to third-party compatibility issues.
- (Optional)** Click **Download as a file for later use**.
- Click **Save** to save the edited barcodes/Bio Sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a file:

1. Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
2. Select the **Barcoded Sample File** you just edited. You see **Success** on the line below, assuming the file is formatted correctly.

Demultiplexed Output Data Set Name (Required)

- Specify the name for the new demultiplexed Data Set that will display in SMRT Link. The utility creates a copy of the input Data Set, renames it to the name specified, and creates demultiplexed child Data Sets linked to it. The input data set remains separate and unmodified.

Parameters

Advanced parameters	Default value	Description
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Minimum Barcode Score	80	A barcode score measures the alignment between a barcode attached to a read and an ideal barcode sequence, and is an indicator of how well the chosen barcode pair matches. It ranges between 0 (no match) and 100 (a perfect match). Specifies that reads with barcode scores below this minimum value are not included in the analysis. This affects the output BAM file and the output demultiplexed Data Set XML file.
Advanced lima Options	NONE	Space-separated list of custom <code>lima</code> options. Not all supported command-line options can be used, and HPC settings cannot be modified. See the Demultiplex Barcodes section of the document SMRT® Tools reference guide v11.1 for information on <code>lima</code> .
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Demultiplex Barcodes utility generates the following reports:

Barcodes > Summary Metrics

- **Unique Barcodes:** The number of unique barcodes in the sequence data.
- **Barcoded HiFi Reads:** The number of correctly-barcoded reads in the HiFi sequence data.
- **Unbarcoded HiFi Reads:** The number of reads in the HiFi sequence data that do not contain barcodes.
- **Barcoded HiFi Read (%):** The percentage of reads in the HiFi sequence data that contain barcodes.

- **Barcoded HiFi Yield (bp):** The number of bases in HiFi sequence data reads that contain barcodes.
- **Unbarcoded HiFi Yield (bp):** The number of bases in HiFi sequence data reads that do not contain barcodes.
- **Barcoded HiFi Yield (%):** The percentage of bases in HiFi sequence data reads that contain barcodes.
- **Unbarcoded HiFi Yield (%):** The percentage of bases in HiFi sequence data reads that do not contain barcodes.
- **Mean HiFi Reads per Barcode:** The mean number of HiFi reads per barcode combination.
- **Max. HiFi Reads per Barcode:** The maximum number of HiFi reads per barcode combination.
- **Min. HiFi Reads per Barcode:** The minimum number of HiFi reads per barcode combination.
- **Barcoded HiFi Read Length (mean, bp):** The mean read length of HiFi reads per barcode combination, in base pairs.
- **Unbarcoded HiFi Read Length (mean, bp):** The mean read length of HiFi reads not containing barcodes, in base pairs.

Barcodes > Barcode Data

- **Sample Name:** The name of the biological sample associated with the barcode combination.
- **Barcode:** A string containing the pair of barcode indices for which the following metrics apply.
- **Barcode Quality:** The barcode quality (QV) associated with the barcode combination.
- **HiFi Reads:** The number of HiFi reads associated with the barcode combination.
- **HiFi Read Length (mean, bp):** The mean read length of HiFi reads per barcode combination, in base pairs.
- **HiFi Read Quality (mean, QV):** The mean barcode quality (QV) associated with the barcode combination.
- **HiFi Yield (bp):** The number of bases in HiFi sequence data reads that contain barcodes.
- **Polymerase Read Length (mean, bp):** The mean read length of polymerase reads associated with the barcode combination, in base pairs.
- **Polymerase Yield (bp):** The number of bases in polymerase reads associated with the barcode combination, in base pairs.

Barcodes > Inferred Barcodes

- **Barcode Name:** The barcode name.
- **Number of Reads:** The number of reads out of the first 35,000 that are inferred to be assigned to the barcode combination.
- **Mean Barcode Score:** The mean barcode score associated with the reads inferred to be associated with the barcode combination.

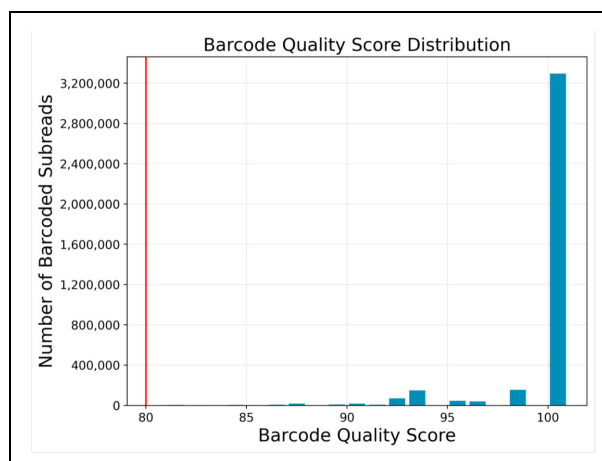
Barcodes > Barcoded Read Statistics

- **Number of Reads per Barcode:** Line graph displays the number of sorted reads per barcode.
 - **Good performance:** The Number of Reads per Barcode line (blue) should be mostly linear. Note that this depends on the choice of Y-axis scale. The mean Number of Reads per Barcode line (red) should be near the middle of the graph and should not be skewed by samples with too many or too few barcodes.

- **Questionable performance:** A sharp discontinuity in the blue line, followed by no yield, with the red line way far from the center. Check the output file **Inferred Barcodes**, note the correct barcodes used, and consider reanalyzing the multiplexed samples with the correct Bio Sample names for the barcodes actually used. If you reanalyze the data, ensure that the **Barcode Name** file includes **only** the correct barcodes used.
- **Barcode Frequency Distribution:** Histogram distribution of read counts per barcode.
 - **Good performance:** A uniform distribution, which is most often a fairly tight symmetric normal distribution, with few barcodes in the tails.
 - **Questionable performance:** A large peak at zero. This can indicate use of incorrect barcodes. Check the output file **Inferred Barcodes**, note the correct barcodes used, and consider reanalyzing the multiplexed samples with the correct Bio Sample names for the barcodes actually used. If you reanalyze the data, ensure that the **Barcode Name** file includes **only** the correct barcodes used.
- **Mean Read Length Distribution:** Histogram distribution of the mean polymerase read length for all samples.
 - **Good performance:** The distribution should be normal with a relatively tight range.
 - **Questionable performance:** A spread out distribution, with a mode towards the low end.

Barcodes > Barcode Quality Scores

- **Barcode Quality Score Distribution:** Histogram distribution of barcode quality scores. The scores range from 0-100, with 100 being a perfect match. Any significant modes or accumulation of scores <60 suggests issues with some of the barcode analyses. The red line is set at 80 – the minimum default barcode score.
 - **Good performance:** HiFi demultiplexing runs should have >90% of reads with barcode quality score ≥ 95 .



- **Questionable performance:** A bimodal distribution with a large second peak usually indicates that some barcodes that were sequenced were **not** included in the barcode scoring set.

Barcodes > Barcoded Read Binned Histograms

- **Read Length Distribution By Barcode:** Histogram distribution of the polymerase read length by barcode. Each column of rectangles is similar to a read length histogram rotated vertically, seen from the top. Each sample should have similar polymerase read length distribution. Non-smooth changes in the pattern looking from left to right might indicate suboptimal performance.
- **Barcode Quality Distribution By Barcode:** Histogram distribution of the per-barcode version of the **Read Length Distribution by Barcode** histogram. The histogram should contain a single cluster of hot spots in each column. All barcodes should also have similar profiles; significant differences in the pattern moving from left to right might indicate suboptimal performance.
 - **Good performance:** All columns show a single cluster of hot spots.
 - **Questionable performance:** A bimodal distribution would indicate missing barcodes in the scoring set.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **All Barcodes (FASTQ):** All barcoded reads, in FASTQ format.
- **User-Input Barcode Samples:** CSV file containing user-entered Bio Sample Name and Barcode names.
- **Barcode Files:** Barcoded subread Data Sets; one file per barcode.
- **Barcoding Summary CSV:** Data displayed in the reports, in CSV format. This includes Bio Sample Name.
- **Barcode Summary:** Text file listing how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- **Inferred Barcodes:** Inferred barcodes used in the analysis. The barcoding algorithm looks at the first 35,000 ZMWs, then selects barcodes with ≥ 10 counts and mean scores ≥ 45 .
- **Unbarcoded Reads:** BAM file containing reads not associated with a barcode.
- **demultiplex.<barcode>.hifi.reads.fastq.gz:** Gzipped HiFi reads in FASTQ format, one file per barcode.

Export Reads Use this utility to export HiFi reads that pass filtering criteria as FASTA, FASTQ and BAM files.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.
- For **barcoded** runs, you must **first** run the **Demultiplex Barcodes** utility to create BAM files **before** using this utility.
- This utility does **not** generate any reports.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Output FASTA File (Default = ON)

- Outputs a single FASTA/FASTQ file containing all the reads that passed the filtering criteria.

Output BAM File (Default = OFF)

- Outputs a single BAM file containing all the reads that passed the filtering criteria.

Min. CCS Predicted Accuracy (Phred Scale) Default = 20

- Phred-scale integer QV cutoff for filtering HiFi reads. The default for **all** applications is 20 (QV 20), or 99% predicted accuracy.

Parameters

Advanced parameters	Default value	Description
Filters to Add to the Data Set	NONE	A semicolon-separated (not comma-separated) list of other filters to add to the Data Set.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **hifi_reads.fasta.gz:** Sequence data that passed filtering criteria, converted to Gzipped FASTA format.

- **hifi_reads.fastq.gz**: Sequence data that passed filtering criteria, converted to Gzipped FASTQ format.
- **<Reads>.bam**: Sequence data that passed filtering criteria, in BAM format.

Mark PCR Duplicates

Use this utility to remove duplicate reads from a **HiFi reads** Data Set created using an ultra-low DNA sequencing protocol.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Note: If starting with a very low-input DNA sample using the **SMRTbell gDNA sample amplification kit**, you **must** run this utility (preceded by the **Trim Ultra-Low Adapters** utility) on the resulting Data Set **prior** to running any secondary analysis application.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Parameters

Advanced parameters	Default value	Description
Identify Duplicates Across Sequencing Libraries	ON	Duplicate reads are identified per sequencing library. The library is specified in the BAM read group LB tag, which is set using the Well Sample Name field in Run Design. By convention, different LB tags correspond to different library preparations. Use this option when the LB tag does not follow this convention to treat all reads as from the same sequencing library.
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Mark PCR Duplicates utility generates the following reports:

PCR Duplicates > Duplicate Rate (table)

- Library:** The name of the library containing duplicate molecules.
- Unique Molecules:** The number of unique molecules in the library.
- Unique Molecules (%):** The percentage of unique molecules in the library.
- Duplicate Reads:** The number of duplicate reads in the library.
- Duplicate Reads (%):** The percentage of duplicate reads in the library.

PCR Duplicates > Duplicate Rate (chart)

- Duplicate Rate:** Displays the percentage of duplicate reads per library.
- Duplicate Reads per Molecule:** Displays the percentage of duplicated molecules per library; broken down by the number of reads per duplicated molecule.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **PCR Duplicates:** BAM file containing duplicate reads with PCR adapters.
- **<Data Set> (deduplicated):** Output Data Set, with duplicate reads with PCR adapters removed.

Read Segmentation

Use this utility to split arrayed HiFi reads at adapter positions, generating segmented reads (S-reads) which are the comprising fragments. For **each** input HiFi read, the utility creates multiple BAM records, one for each fragment. An arrayed HiFi read can contain many fragments.

- The utility accepts **HiFi reads** (BAM/Data Set format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Segmentation Adapter Set

- Specify a FASTA file, provided by PacBio, containing segmentation adapters. If you need a **custom** segmentation adapter set, click **Advanced Parameters** and use a custom FASTA file formatted as described below.

Parameters

Advanced parameters	Default value	Description
Adapters FASTA	NONE	Specify a custom FASTA file containing segmentation adapters. If not specified, the adapters specified in the XML metadata are used. Adapters must be ordered in the expected order of adapters in the reads. There should be one entry per adapter (forward or reverse-complement orientation) with no overlapping adapter sequences. Duplicate names or sequences are not allowed. Example: >A AGCTTACTTGTGAAGA >B ACTTGTAAGCTGTCTA >C ACTCTGTCAGGTCCGA >D ACCTCCTCCTCCAGAA >E AACCGGACACACTTAG
Base task memory (MB)	512	Specifies the minimum memory per task when submitting a job.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Read Segmentation utility generates the following reports:

Read Segmentation > Summary Metrics

- **Reads:** The number of input arrayed HiFi reads.
- **Segmented reads (S-reads):** The number of generated S-reads.
- **Mean length of S-reads:** The mean read length of the generated S-reads.
- **Percent of reads with full arrays:** The percentage of input reads containing all adapter sequences in the order listed in the segmentation adapter FASTA file.
- **Mean array size:** The mean number of fragments (or S-reads) found in the input reads.

Read Segmentation > Segmentation Statistics

- Histogram distribution of the number of S-reads per read.
- Heatmap of adapter ligations.

Read Segmentation > Length of Reads

- Histogram distribution of the number of HiFi reads by read length, in base pairs.

Read Segmentation > S-read Length Distribution

- Histogram distribution of the number of S-reads by the HiFi read length, in base pairs.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Report read_segmentation:** JSON report containing summary statistics.
- **Segmented Reads, passing, unaligned:** BAM file containing the generated S-reads that passed filtering.
- **Non-passing reads, unaligned:** BAM file containing HiFi reads that did **not** generate S-reads.
- **<Data Set> Segmented Reads:** Output Data Set, containing generated S-reads and supplementary files.

Trim Ultra-Low Adapters

Use this utility to trim PCR Adapters from a HiFi reads Data Set created using an ultra-low DNA sequencing library.

- The utility accepts **HiFi reads** (BAM format) as input. **HiFi reads** are reads generated with CCS analysis whose quality value is equal to or greater than 20.

Note: If starting with a very low-input DNA sample using the **SMRTbell gDNA sample amplification kit**, you **must** run this utility (followed by the **Mark PCR Duplicates** utility) on the resulting Data Set **prior** to running any secondary analysis application.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

PCR Adapters (Required)

- Specify the file of PCR adapters used during library preparation of an ultra-low DNA sequencing library to be trimmed from the sequenced data.

Parameters

Advanced parameters	Default value	Description
Min. CCS Predicted Accuracy (Phred Scale)	20	Phred-scale integer QV cutoff for filtering HiFi reads. The default for all applications is 20 (QV 20), or 99% predicted accuracy.
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Trim Ultra-Low Adapters utility generates the following reports:

PCR Adapters > Summary Metrics

- **Unique PCR Adapters:** The number of unique PCR adapters in the sequence data.
- **HiFi Reads With PCR Adapters:** The number of reads in the sequence data that contain PCR adapters.
- **HiFi Reads Without PCR Adapters:** The number of reads in the sequence data that do **not** contain PCR adapters.
- **Percent Reads with Adapters:** The percentage of reads in the sequence data that contain PCR adapters.
- **Percent Bases in Reads with Adapters:** The percentage of bases in reads in the sequence data that contain PCR adapters.
- **Mean HiFi Reads Per Adapter:** The mean number of reads per PCR adapter in the sequence data.

- **Max. HiFi Reads Per Adapter:** The maximum number of reads per PCR adapter in the sequence data.
- **Min. HiFi Reads Per Adapter:** The minimum number of reads per PCR adapter in the sequence data.
- **Mean HiFi Read Length:** The mean read length of reads per PCR adapter in the sequence data.

PCR Adapters > PCR Adapter Data

- **Bio Sample Name:** The name of the biological sample associated with the PCR adapters.
- **PCR Adapter Name:** A string containing the pair of PCR adapter indices for which the following metrics apply.
- **Mean PCR Adapter Quality:** The mean PCR adapter quality associated with the PCR adapter.
- **HiFi Reads:** The number of HiFi reads associated with the PCR adapter.
- **HiFi Read Length (mean, bp):** The mean read length of HiFi reads associated with the PCR adapter.
- **HiFi Yield (bp):** The total yield (in base pairs) of the HiFi reads associated with the PCR adapter.

PCR Adapters > PCR Adapter Read Statistics

- **Number of Reads Per PCR Adapter:** Histogram distribution of the mean number of reads per PCR adapter.
- **PCR Adapter Frequency Distribution:** Histogram distribution of reads with PCR adapter mapped to the number of barcoded samples.
- **Mean Read Length Distribution:** Maps the mean read length against the number of barcoded samples.

PCR Adapters > PCR Adapter Quality Scores

- Histogram distribution of PCR adapter quality scores. The scores range from 0-100, with 100 being a perfect match.

PCR Adapters > PCR Adapter Read Binned Histograms

- **Read Length Distribution By PCR Adapter:** Histogram distribution of the read length by PCR adapter. Each column of rectangles is similar to a read length histogram rotated vertically, seen from the top.
- **PCR Adapter Quality Distribution By Barcode:** Histogram distribution of the per-barcode version of the **Read Length Distribution by PCR Adapter** histogram.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **Reads Missing Adapters:** Reads Missing Adapters: BAM file containing the reads with missing PCR adapters from the input Data Set.
- **PCR Adapter Data CSV:** Includes the data displayed in the PCR Adapter Data table.
- **<Data Set> (trimmed):** Output Data Set, with the PCR adapters removed.

Circular Consensus Sequencing (CCS)

Use this utility to identify consensus sequences for single molecules.

- The utility accepts **Subreads** (BAM format) as input.

Importing/exporting analysis settings

- Click **Import Analysis Settings** and select a previously-saved CSV file containing the desired settings (including **Advanced Parameters**) for the selected utility. The imported utility settings are set.
- Click **Export** to create a CSV file containing all the settings you specified for the utility. You can then import this file when creating future jobs using the same utility. You can also use this exported file as a template for use with later jobs.

Detect 5mC Sites (Default = OFF)

- If set to **ON**, kinetics analysis to identify 5mC CpG sites will be performed.

Parameters

Advanced parameters	Default value	Description
Minimum CCS Read Length	10	The minimum length for the median size of insert reads to generate a consensus sequence. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates.
Maximum CCS Read Length	50,000	The maximum length for the median size of insert reads to generate a consensus sequence. If the targeted template is known to be a particular size range, this can filter out alternative DNA templates.
Generate a Consensus for Each Strand	OFF	Generate a consensus for each strand. Warning: This is an experimental option for the CCS algorithm, and may not be compatible with all downstream applications. We recommend using command-line analysis for this feature.
Process All Reads	OFF	Specifies behavior identical to on-instrument CCS reads generation, overriding all other cutoffs. This setting writes a CCS read for every ZMW in the input Data Set. Set to OFF to specify more restrictive settings.
Include Kinetics information with CCS Analysis output	OFF	If ON , include kinetics per-base data required for methylation DNA analysis. Note: This results in a BAM file that is 3-4 times larger. This option applies only when Process All Reads is set to ON .
Advanced CCS Options	NONE	Space-separated list of additional command-line options to CCS analysis. Not all supported command-line options can be used, and HPC settings cannot be modified. See SMRT® Tools reference guide v11.1 for details.
Minimum Predicted Accuracy (Deprecated)	0.99	The minimum predicted accuracy of a read, ranging from 0 to 1. (0.99 indicates that only reads expected to be 99% accurate are emitted.) Note: This setting is ignored if the Process All Reads advanced parameter is set to ON .
Minimum Number of Passes (Deprecated)	3	The minimum number of full passes for a ZMW to be used. Full passes must have an adapter hit before and after the insert sequence and so do not include any partial passes at the start and end of the sequencing reaction. Note: This setting is ignored if the Process All Reads advanced parameter is set to ON .
Detect And Split Heteroduplex Read	OFF	Specifies that any detected heteroduplexes are separated into separate reads.

Advanced parameters	Default value	Description
Compute Settings	Select	(Optional) Specify the distributed computing cluster settings configuration, if made available by the site SMRT Link administrator.

Reports and data files

The Circular Consensus Sequencing (CCS) utility generates the following reports:

CCS Analysis Report > Summary Metrics

Note: CCS reads with quality value equal to or greater than 20 are called **HiFi reads**.

- **HiFi Reads:** The total number of CCS reads whose quality value is equal to or greater than 20.
- **HiFi Yield (bp):** The total yield (in base pairs) of the CCS reads whose quality value is equal to or greater than 20.
- **HiFi Read Length (mean, bp):** The mean read length of the CCS reads whose quality value is equal to or greater than 20.
- **HiFi Read Quality (median):** The median number of CCS reads whose quality value is equal to or greater than 20.
- **HiFi Number of Passes (mean):** The mean number of passes used to generate CCS reads whose quality value is equal to or greater than 20.

CCS Analysis Report > HiFi Read Length Summary

- **Read Length (bp):** The HiFi read length, ranging from ≥ 0 to $\geq 40,000$ base pairs.
- **Reads:** The number of HiFi reads with the specified read length.
- **Reads (%):** The percentage of HiFi reads with the specified read length.
- **Yield (bp):** The number of base pairs in the HiFi reads with the specified read length.
- **Yield (%):** The percentage of base pairs in the HiFi reads with the specified read length.

CCS Analysis Report > HiFi Read Quality Summary

- **Read Quality (Phred):** Phred-scale quality values, ranging from QV ≥ 20 to QV ≥ 50 .
- **Reads:** The number of HiFi reads with the specified read quality.
- **Reads (%):** The percentage of HiFi reads with the specified read quality.
- **Yield (bp):** The number of base pairs in the HiFi reads with the specified read quality.
- **Yield (%):** The percentage of base pairs in the HiFi reads with the specified read quality.

CCS Analysis Report > Read Length Distribution

- **HiFi Read Length Distribution:** Histogram distribution of HiFi reads by read length.
- **Yield by HiFi Read Length:** Histogram distribution of the cumulative yields of CCS reads by read length.
- **Read Length Distribution:** Histogram distribution of all reads by read length.

CCS Analysis Report > Number of Passes

- Histogram of the number of complete subreads in CCS reads, broken down by number of reads.

CCS Analysis Report > Read Quality Distribution

- Histogram distribution of the CCS reads by the Phred-scale read quality.

CCS Analysis Report > Predicted Accuracy vs. Read Length

- Heat map of CCS read lengths and predicted accuracies.

Data > File Downloads

The following files are available on the analysis results page. Additional files are available on the SMRT Link server, in the analysis output directory.

- **Analysis Log:** Log information for the analysis execution.
- **SMRT Link Log:** Server-level analysis log information. (This file is displayed when you choose **Data > SMRT Link Log**.)
- **CCS Analysis Per-Read Details:** Summary of CCS analysis performance and yield.
- **hifi_reads.fastq.gz:** Gzipped HiFi reads in FASTQ format.
- **hifi_reads.fasta.gz:** Gzipped HiFi reads in FASTA format.
- **hifi_reads.bam:** HiFi reads in BAM format.
- **All Reads (BAM):** BAM file containing one CCS read per ZMW, including the following types of reads:
 - HiFi reads (Q20 or higher)
 - Lower-quality but still polished consensus reads (Q1-Q20)
 - Unpolished consensus reads (RQ=-1)
 - 0- or 1-pass subreads unaltered (RQ=-1)

Working with barcoded data

This section describes how to use SMRT Link to work with barcoded data. Demultiplex Barcodes analysis is powered by the `lima` SMRT Analysis tool.

The canned data provided with SMRT Link v11.1 includes 9 barcode sets:

1. gDNA Amplification Adapter
2. Iso-Seq 12 Barcoded cDNA Primers
3. Iso-Seq cDNA Primers
4. Barcoded Overhang Adapter Kits 8A and 8B
5. Barcoded M13 Primer Plate
6. SMRTbell Barcoded Adapter Plate 3.0
7. Mas-Seq Adapter v1 (MAS16)
8. Mas-Seq Adapter v2 (MAS12)
9. 10x Chromium single cell 3' cDNA primers

SMRT Link v11.1 supports sample traceability through the various modules in the application by using the **Bio Sample Name**.

Run Design in SMRT Link v11.1 contains a required **Bio Sample Name** field for both single and multiplexed samples.

- For multiplexed experiments, SMRT Link provides default names for **one** Bio Sample Name per barcode, which can be edited as needed in the **Barcoded Sample Names** file.
- For multiplexed Iso-Seq Analysis **only**, Bio Sample Names are **not** required.

Well Sample Name and **Bio Sample Name** entered in Sequel II systems, and in Run Designs for multiplexed runs:

- Display as column values in the Data Management and SMRT Analysis modules.
- Display as Data Set attributes in the Data Set details page in Data Management.
- Populate the `LB` and `SM` tags in read group headers of BAM files containing basecalled data.

Example Well Sample Names and Bio Sample Names

- **Non-barcoded Well Sample Name:** HG002_2019_11_2_10K
- **Non-barcoded Bio Sample Name:** HG002
- **Barcoded Well Sample Name:** My Multiplexed Set of Bugs
- **Barcoded Bio Sample Name:**
Unknown Microbe 1,...,Unknown Microbe N

Step 1: Specify the barcode setup and sample names in a Run Design

Note: If you specified the barcode setup in Run Design, the demultiplexing is performed **automatically** after the data is transferred to the SMRT link server. You can also specify the barcode setup **manually** by selecting **SMRT Analysis > Create New Job** and then selecting the Demultiplex Barcodes data utility.

1. In SMRT Link, create a new Run Design as described in [“Creating a new Run Design” on page 16](#). **Before** you finish the new Run Design, perform the following steps.

Barcoded Sample Options

Sample Is Barcoded ☒ YES ☐ NO

Barcode Set Required Sequel_16_barcode_v3

Same Barcodes on Both Ends of Sequence ☒ YES ☐ NO

Assign Bio Sample Names to Barcodes Required

Autofilled Barcoded Sample File

Barcoded Sample Name File Required

2. Click **Barcoded Sample Options** and then click **Yes** for **Sample is Barcoded**. Additional fields related to barcoding display.
3. Specify a **Barcode Set** using the dropdown list.
Note: You can specify up to 10,000 samples. Specifying **more** than 10,000 samples may cause a delay of several minutes in analysis submission.
4. Specify if the **same** barcodes are used on both ends of the sequences.
 - Selecting **Yes** specifies symmetric and tailed designs where **all** the reads have the same barcodes on both ends of the insert sequence. Barcode analysis of such experiments retains **only** data with the same barcode identified on both ends.
 - Selecting **No** specifies asymmetric designs where the barcodes are **different** on each end of the insert. Barcode analysis of such experiments retains any barcode pair combination identified in the Data Set.
5. SMRT Link automatically creates a CSV-format **Autofilled Barcode Name** file. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names 1 through N, corresponding to barcodes 1 through N, for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names

to barcodes. (**Note:** Bio Sample Names are hardcoded and can be traced through secondary analysis using SMRT Analysis.)

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the check boxes to select multiple barcodes.)
- **(Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid using spaces in Bio Sample Names as they may lead to third-party compatibility issues.
- **(Optional)** Click **Download as a file for later use**.
- Click **Save** to save the edited barcodes/Bio Sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a File:

- Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
 - Select the **Barcoded Sample** file you just edited. You see **Success** on the line below, assuming the file is formatted correctly.
6. Specify **if** and **where** to automatically generate **HiFi reads** (reads generated with CCS analysis whose quality value is equal to or greater than 20):
- **On Instrument** (available **only** for the Sequel IIe system): HiFi reads are automatically generated on the instrument, **before** transfer to the compute cluster where SMRT Link is installed.
 - **In SMRT Link:** HiFi reads are automatically generated **after** transfer to the compute cluster where SMRT Link is installed.
 - **Do Not Generate:** HiFi reads are **not** generated for this run. Only subread data are transferred to the local compute cluster where SMRT Link is installed.
7. Click **Save**.

Step 2: Perform the sequencing run

Load the samples and perform the sequencing run, using the Run Design you created in Step 1. The demultiplexing analysis is performed automatically on the SMRT Link server once the data is transferred from the instrument. This creates an analysis of type `Demultiplex Barcodes (Auto)` in the SMRT Analysis module. You can click to select this analysis and review the reports and data created. If everything looks fine, you can continue to **Step 4** and use the demultiplexed Data Set(s) created by the run as input to further analysis.

Note: By default, `Demultiplex Barcodes (Auto)` creates **one** Data Set per autodetected barcode within the selected barcode set. It also applies

a Data Set filter of a minimum barcode score greater than 26 for optimal results in secondary analysis. If used, the analysis parameter **Filters to add to the Data Set** overrides other barcode filtering, even if the barcode score set with it is lower than 26.

**Step 3:
(Optional) Run
the Demultiplex
Barcodes data
utility**

If instead you did **not** specify the barcode setup in the Run Design, or if you need to change any of the parameters used in the `Demultiplex Barcodes` analysis automatically launched from Run Design, run the **Demultiplex Barcodes** data utility. This separates reads by barcode and creates a new demultiplexed Data Set that you can then use as input to other secondary analysis applications.

1. Click **+ Create New Job**.
2. Enter a **name** for the job.
3. Select **Data Utility** as the workflow type.
4. Select **HiFi reads** as the data type to use. The Data Sets table displays the appropriate Data Sets available for the job.
5. In the Data Sets table, select one or more Data Sets to be analyzed together.
6. Click **Next**.
7. Select **Demultiplex Barcodes** from the Applications list.
8. Specify a **Barcode Set** (barcode sequence file.)
Note: You can specify up to 10,000 samples. Specifying **more** than 10,000 samples may cause a delay of several minutes in analysis submission.
9. Specify if the **same** barcodes are used on both ends of the sequences.
 - Selecting **Yes** specifies symmetric and tailed designs where **all** the reads have the same barcodes on both ends of the insert sequence. Barcode analysis of such experiments retains **only** data with the same barcode identified on both ends.
 - Selecting **No** specifies asymmetric designs where the barcodes are **different** on each end of the insert. Barcode analysis of such data retains any barcode pair combination identified in the Data Set.
10. SMRT Link automatically creates a CSV-format **Autofilled Barcoded Sample** file. The barcode name is populated based on your choice of barcode set, and if the barcodes are the same at both ends of the sequence. The file includes a column of automatically-generated Bio Sample Names **1** through **N**, corresponding to barcodes **1** through **N**, for the biological sample names. There are **two different ways** to specify which barcodes to use, and assign biological sample names to barcodes:

Interactively:

- Click **Interactively**, then drag barcodes from the **Available Barcodes** column to the **Included Barcodes** column. (Use the check boxes to select multiple barcodes.)
- **(Optional)** Click a Bio Sample field to edit the Bio Sample Name associated with a barcode. **Note:** Avoid using spaces in Bio Sample Names as they may lead to third-party compatibility issues.

- (Optional) Click **Download as a file for later use**.
- Click **Submit** to save the edited barcodes/bio sample names. You see **Success** on the line below, assuming the file is formatted correctly.

From a File:

- Click **From a File**, then click **Download File**. Edit the file and enter the biological sample names associated with the barcodes in the second column, then save the file. Use alphanumeric characters, spaces (allowed but **not recommended** for compatibility with third-party downstream software), hyphens, underscores, colons, or periods **only** - other characters will be removed **automatically**, with a maximum of 40 characters. If you did **not** use all barcodes in the Autofilled Barcode Name file in the sequencing run, **delete** those rows.
 - **Note:** Open the CSV file in a text editor and check that the columns are separated by **commas**, not semicolons or tabs.
 - Select the **Barcoded Sample** file you just edited. You see **Success** on the line below, assuming the file is formatted correctly.
11. Specify the **name** for the new demultiplexed Data Set that will display in SMRT Link. The application creates a copy of the input Data Set, renames it to the name specified, and creates demultiplexed child Data Sets linked to it. The input Data Set remains separate and unmodified.
 12. (Optional) Specify any advanced parameters.
 13. Click **Start**. After the analysis is finished, a new demultiplexed Data Set is available.

Note: For information about the reports generated by the Demultiplex Barcodes data utility, see [“Demultiplex Barcodes” on page 99](#).

Step 4: Run applications using the demultiplexed data as input

All secondary analysis applications except **Demultiplex Barcodes** can use demultiplexed Data Sets as input.

Note: For **Iso-Seq** analysis with barcoded samples, use the Iso-Seq application **instead** of the Demultiplex Barcodes data utility, as the Iso-Seq application **already** includes the demultiplexing step as part of the pipeline. When performing multiplexed Iso-Seq analysis, ensure that the Run Design **Sample Is Barcoded** option is set to **No** (the default setting). Then, in SMRT Analysis, go straight to the Iso-Seq application and, in the parameters section, select a Primer Set containing multiple primers, such as `IsoSeq_Primers_12_Barcodes_v1`.

1. Select the secondary analysis application/data utility to use.
2. Click the number in the **Demultiplexed Subsets** column, then select the demultiplexed Data Set to use as input:

Data Sets

<input type="checkbox"/>	Name	Demultiplexed Subsets
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles	3

Members of Re-barcode Alice/Bob/Charles

<input type="checkbox"/>	Name	Well Sample Name
<input checked="" type="checkbox"/>	Re-barcode Alice/Bob/Charles (Charles)	T0213_384-plex_barcode_A01
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles (Bob)	T0213_384-plex_barcode_A01
<input type="checkbox"/>	Re-barcode Alice/Bob/Charles (Alice)	T0213_384-plex_barcode_A01

– You can select the **entire** Data Set as input, or one or more specific outputs from selected barcodes, to a maximum of 16 sub-Data Sets, 12 for Iso-Seq.

3. Additional **Analysis Type** options become available. You can select from the following options:

Workflow Type

☒ ANALYSIS ☐ AUTO ANALYSIS ☐ DATA UTILITY

Analysis of Multiple Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis for All Data Sets

One Analysis per Data Set - Identical Parameters

One Analysis per Data Set - Custom Parameters

- **One Analysis for All Data Sets:** Runs one job using all the selected barcode Data Sets as input, for a maximum of 30 Data Sets.
- **One Analysis per Data Set - Identical Parameters:** Runs **one** separate job for **each** of the selected barcode Data Sets, using the **same** parameters, for a maximum of 10,000 Data Sets. Optionally click **Advanced Parameters** and modify parameters.
- **One Analysis per Data Set - Custom Parameters:** Runs **one** separate job for **each** of the selected barcode Data Sets, using **different** parameters for each Data Set, for a maximum of 16 Data Sets. Click **Advanced Parameters** and modify parameters. Then click **Start and Create Next**. You can then specify parameters for each of the included barcode Data Sets.
- **Note:** The number of Data Sets listed is based on testing using PacBio's suggested compute configuration, listed in **SMRT Link software installation guide (v11.1)**.

4. Click **Start** to submit the job.

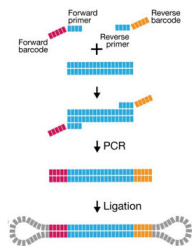
Demultiplex Barcodes details

The **Demultiplex Barcodes** data utility identifies barcode sequences in PacBio single-molecule sequencing data.

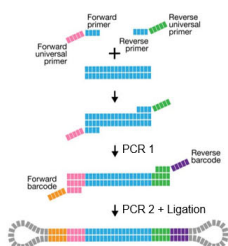
Demultiplex Barcodes can demultiplex samples that have a unique per-sample barcode pair and were pooled and sequenced on the same SMRT Cell. There are four different methods for barcoding samples with PacBio technology:

- 1. Barcoded target-specific primers
- 2. Barcoded universal primers
- 3. Barcoded overhang adapters
- 4. Barcoded linear adapters (target capture)

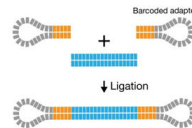
1. Barcoded Target-Specific Primers



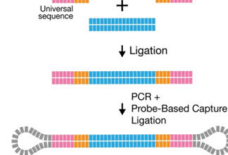
2. Barcoded Universal Primer



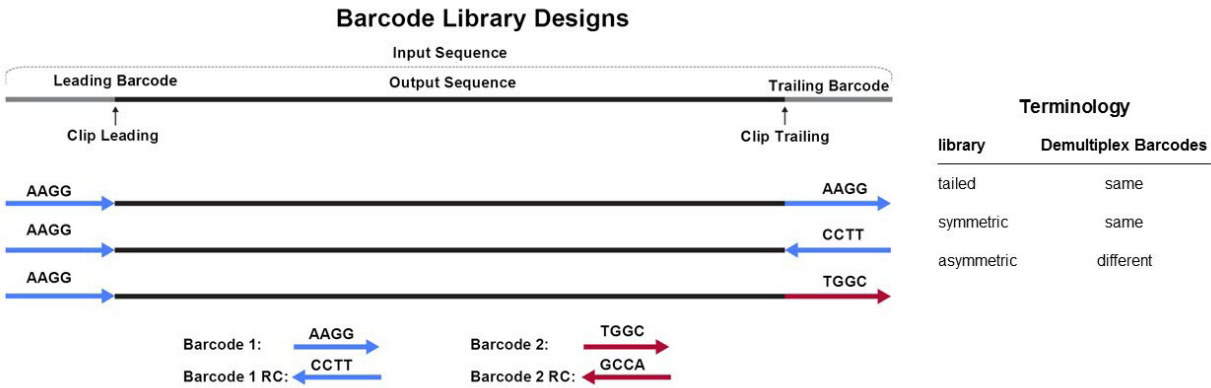
3. Barcoded Overhang Adapters



4. Barcoded Linear Adapter (Target Capture)



In addition, there are three different barcode library designs.



The **Demultiplex Barcodes** application in SMRT Link supports demultiplexing of subreads.

Demultiplexing of CCS reads is possible using the command line. (See **SMRT® Tools reference guide (v11.1)** for details.)

Symmetric mode

For **symmetric** and **tailed** library designs, the **same** barcode is attached to both sides of the insert sequence of interest. The only difference is the orientation of the trailing barcode. For barcode identification, one read with a single barcode region is sufficient. Symmetric barcoding is used for samples constructed using Barcoded overhang adapters, Barcoded universal primer and target enrichment (linear). This is also the default scoring mode in SMRT Link v10.2 and later.

Asymmetric mode

Barcode sequences are **different** on the ends of the SMRTbell template. Asymmetric mode is used with the M13 barcoding procedure. (See the document **Procedure & checklist - Preparing SMRTbell libraries using PacBio barcoded M13 primers for multiplex SMRT sequencing** for details.) PacBio using this mode **only** for small inserts (up to 5 kb) where both ends of the insert are expected to be sequenced. **Both** barcodes must be detected.

Note: For **both** Symmetric and Asymmetric modes, the limit for unique individual barcode sequences is 768, and the limit for the number of different barcode pairs is 10,000.

When running the **Demultiplex Barcodes** data utility in SMRT Link, set the **Same Barcodes on Both Ends of the Sequence** option to **Off**.

Mixed mode

Libraries with combined symmetric and asymmetric barcoding are **not** supported.

Automated analysis

Auto Analysis and **Pre Analysis** allow a specific analysis to be **automatically** run after a sequencing run has finished and the data is transferred to the SMRT Link server. The analysis can include demultiplexed output.

- Auto Analysis can be set up in Run Design or SMRT Analysis **after** the Run Design is saved and **before** the run is loaded on the instrument.
- Auto Analysis can be run on HiFi reads, and includes **all** secondary analysis applications available.
- Auto Analysis works with **all** Sequel II systems and Sequel IIe systems.

Pre Analysis is the process of CCS analysis and/or demultiplexing of Sequel basecalled data. Pre Analysis occurs **before** Auto Analysis, and is defined when you create a Run Design and specify one or more of the following:

- Read Type = **HiFi reads** and Generate HiFi reads = **On Instrument** or **In SMRT Link**.
- Read Type = **HiFi reads** and Sample is Barcoded = **Yes**.

Note: Pre Analysis works with **all** Sequel II systems and Sequel IIe systems.

Creating an Auto Analysis job from SMRT Analysis

This procedure includes only the basic steps - for more detailed information on creating jobs, see [“Creating and starting a job” on page 44](#).

1. Access SMRT Link using the Chrome web browser.
2. Select **SMRT Analysis**.
3. Click **+ Create New Job**.
4. Enter a **name** for the job.
5. Specify **Auto Analysis** as the workflow type. The table displays available runs. (**Note:** Runs display here **only** if they are in the **Created** state - not if they are already running or have completed.)
6. Click a **Collections** link associated with a specific run in the table.
7. Select **one** sample, or drill down further and select from the barcoded samples of a single collection by clicking the **Barcoded Samples** link. You can select **multiple** barcoded samples at this level.
Note: You **cannot** select a mix of samples and barcoded samples, or select barcoded samples from **multiple** samples. In addition, samples or barcoded samples will **not** be selectable if they have a Job Id.
8. Click **Next**.
9. Select a secondary analysis application from the dropdown list.
Note: **Only** secondary analysis applications, **not** data utilities, are available for use with Auto Analysis.

10. **(Optional)** Click **Advanced Parameters** and specify the values of the parameters to change. Click **OK** when finished. (Different applications have different advanced parameters.)
 - To see information about parameters for **all** secondary analysis applications provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#).
11. Click **Start** to submit the Auto Analysis job.

Creating Auto Analysis from a Run Design

1. Create a new Run Design (See [“Creating a new Run Design” on page 16](#) for details) and save it. The **Auto Analysis** button is enabled only **after** you save the Run Design.
2. Click **Auto Analysis**. This takes you into SMRT Analysis, where you create the new job that will be associated with the collection.
3. Name the new job.
4. Click the numbered **Collections** link (Column 2 of the Runs table) associated with the run that you defined in Step 1. (**Note:** Runs display here **only** if they are in the **Created** state - not if they are already running or have completed.)
5. Select a collection for analysis.
6. Click **Next**.
7. Select a secondary analysis application to use for the analysis.
8. **(Optional)** Click **Advanced Parameters** and specify the values of the parameters to change. Click **OK** when finished. To see information about parameters for **all** secondary analysis applications provided by PacBio, see [“PacBio® secondary analysis applications” on page 54](#).
9. Click **Create**.

HiFiViral SARS-CoV-2: Creating Auto Analysis in Run Design

The **HiFiViral SARS-CoV-2** Application includes a streamlined version of Auto Analysis as part of creating a Run Design.

1. In Run Design, click **Create New Design**.
2. From the Application list, select **HiFiViral SARS-CoV-2**. Preloaded default values display in green.
3. Enter the **Well Sample Name**.
4. By default, **Auto Analysis** is set to **Yes** for **Automatic Launch of SARS-CoV-2 Analysis**.
5. Enter the **Analysis Name**.
6. By default, **Yes** is selected for **Sample Is Barcoded** and **No** for **Same Barcode on Both End of Sequence**.
7. By default, the barcode set **HiFiViral SARS-CoV-2 M13barcodes** is selected.
8. For **Assign Bio Sample Names to Barcodes**, select **From a File** and then click **Download File**.
9. Open the downloaded file (**assayPlateQC_template_4by96.csv**) in a text editor. See [“HiFiViral SARS-CoV-2 Analysis” on page 54](#) for details on modifying this file for your samples.

Getting information about analyses created by Auto Analysis

There are several ways to obtain information on the state of an analysis created using the Auto Analysis feature.

From SMRT Analysis:

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. To filter the jobs, click the funnel in the **State** column header, then click **Created**. This displays **only** jobs in the **Created** state.
3. Click the job of interest.
4. Click **Analysis Overview > Status of Individual Analyses**. This displays information about the analysis, including the application used.

From Run Design:

1. On the home page, select **Run Design**.
2. Click the Run Design of interest.
3. Scroll all the way to the right in the table. This displays information about the samples included in the run, including Pre Analysis and Auto Analysis details.

Visualizing data using IGV

Once an analysis has successfully completed, visualize the results using the **Integrative Genomics Viewer (IGV)**.

- See [here](#) for further installation instruction and usage details.
- See [here](#) for PacBio-specific settings and visualizations.

You can visualize data generated by the following secondary analysis applications:

- Iso-Seq Analysis
- HiFi Mapping
- Microbial Genome Analysis
- Structural Variant Calling

IGV requires the following files for visualization:

- One consolidated alignment BAM file
- BAM index file
- Genome reference file

If an analysis generates **multiple** alignment BAM files, those files must **first** be combined into **one** consolidated alignment BAM file for visualization with IGV.

To visualize data using IGV

1. Create and run the analysis.
2. After the analysis has finished successfully, go to the **Data > IGV Visualization Files** section of the analysis report page.
3. Open IGV and select the reference genome used for the analysis. (See [here](#) for instructions on how to load a genome.)
4. Copy a BAM file link from the **Data > IGV Visualization Files** section of the analysis report page.

Note: If you are performing *de novo* assembly, you **must** use links to the **draft** assembly BAM files, which are clearly labeled.

IGV Visualization Files

File	Path	Size	Type
Draft Assembly	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-collect_configs/execution/collected_ctg.fasta	20 KB	Fasta
Mapped BAM	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-auto_consolidate_alignments/execution/mapped.bam	470 KB	bam
Mapped BAM Index	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-6a4c5fc40434/call-auto_consolidate_alignments/execution/mapped.bam.bai	336 bytes	bam_bai
Draft Assembly Index	http://smrtlink-bihourly.nanofluidics.com:8080/job-data/pb_assembly_microbial/b0d30c74-9c71-40a8-8ae9-fad5cbaf8543cbe1e94fb4dcdbed56/collected_ctg.fasta.fai	107 bytes	SamIndex

Click to copy the file path to the Clipboard.

5. In IGV, choose **File > Load from URL...** and paste the link into the File URL input field. Click **OK**.
6. Repeat for the remaining links.

If you ran an analysis and there are **no Data > IGV Visualization Files** links, the analysis generated multiple alignment BAM files, but did **not** consolidate the files. Click the **Launch BAM Consolidation** button to consolidate them.

IGV Visualization Files

No IGV Visualization Files were found

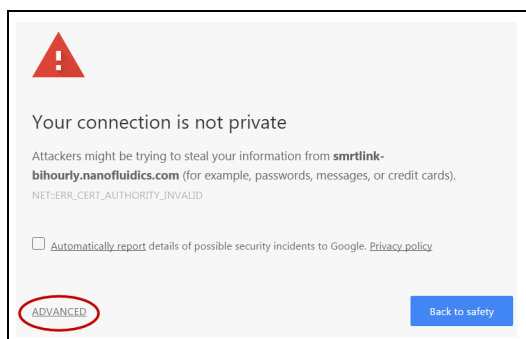
[Launch BAM Consolidation](#)

File	Path	Size	Type
Variants VCF	http://ip-172-32-0-82.us-west-1.compute.internal:9090/job-data/pb_sat/8f517e88-d47e-4250-beda-f6f04ea57894/call-pl_resequencing/job_resequencing/Sat2020a8-8a08-4163-677f-6a4d490cfd6d/call-consensus/consensusu/8627b248-da29-433a-9a35-3f5d474052dc/call-gather_vcf/execution/variants.vcf	583 bytes	vcf

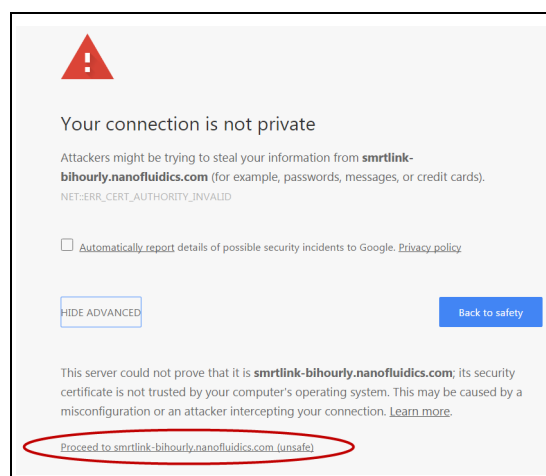
Using the PacBio® self-signed SSL certificate

SMRT Link v11.1 ships with a PacBio self-signed SSL certificate. If this is used at your site, security messages display when you try to login to SMRT Link for the **first time** using the Chrome browser. These messages may also display **other times** when accessing SMRT Link.

1. The first time you start SMRT Link after installation, you see the following. Click the **Advanced** link.



2. Click the **Proceed...** link. (You may need to scroll down.)



3. Close the window by clicking the **Close** box in the corner.



The **Login** dialog displays, where you enter the User Name and Password. The next time you access SMRT Link, the Login dialog displays **directly**.

Sequel® II system and Sequel IIe system output files

This section describes the data generated by the Sequel IIe system and Sequel II system for each SMRT Cell transferred to network storage.

Sequel IIe system output files

Following is a sample of the file and directory structure output by the **Sequel IIe** system (**not** including low-quality reads):

```
<your_specified_output_directory>/r64012_211206ee_183753/1_A01/
|--m64012ee_211206_183753.baz2bam_1.log
|--m64012ee_211206_183753.ccs.log
|--m64012ee_211206_183753.ccs_reports.json
|--m64012ee_211206_183753.ccs_reports.txt
|--m64012ee_211206_183753.consensusreadset.xml
|--m64012ee_211206_183753.hifi_reads.bam
|--m64012ee_211206_183753.hifi_reads.bam.pbi
|--m64012ee_211206_183753.sts.xml
|--m64012ee_211206_183753.zmw_metrics.json.gz
|--m64012ee_211206_183753.transferdone
```

If **5mC CpG Detection** is performed, the following additional files are output:

```
|-- m64012ee_211206_183753.5mc_report.json
|-- m64012ee_211206_183753.primrose.log
```

If **on-instrument demultiplexing** is performed, the following additional files are output. **Note:** The undemultiplexed `hifi_reads.bam` file is **not** transferred; it is partitioned into the file structure shown below:

```
|-- bc1001--bc1001/m64012e_211206_183753.bc1001--bc1001.consensusreadset.xml
|-- bc1001--bc1001/m64012e_211206_183753.hifi_reads.bc1001--bc1001.bam
|-- bc1001--bc1001/m64012e_211206_183753.hifi_reads.bc1001--bc1001.bam.pbi
|-- m64012e_211206_183753.barcodes.fasta
|-- m64012e_211206_183753.lima.log
|-- m64012e_211206_183753.lima_counts.txt
|-- m64012e_211206_183753.lima_guess.json
|-- m64012e_211206_183753.lima_guess.txt
|-- m64012e_211206_183753.lima_reports.txt
|-- m64012e_211206_183753.lima_summary.txt
|-- m64012e_211206_183753.unbarcoded.consensusreadset.xml
|-- m64012e_211206_183753.unbarcoded.hifi_reads.bam
|-- m64012e_211206_183753.unbarcoded.hifi_reads.bam.pbi
```

In these examples:

- `r64012ee_211206_183753` is a directory containing the output files associated with **one** run.
- `r64012ee` is the instrument ID number.

The run directory includes a subdirectory for each collection/cell associated with a sample well - in this case `1_A01`. The collection/cell subdirectory can include the following output files:

- `ccs.log`: Log file from the CCS analysis. Informative for debugging and performance tracking by PacBio.
- `ccs_reports.json`, `ccs_reports.txt`: Contains processing metrics summarizing how many ZMWs generated HiFi reads, and how many ZMWs failed to generate CCS reads. These files contain the same information, and are used internally by PacBio Technical Support.
- `hifi.reads.bam`: Contains the HiFi reads in BAM format.
Note: If low-quality reads are included in the Run Design, the Sequel IIe system will output a `reads.bam` file, which contains **HiFi reads and non-HiFi reads**:
 - HiFi reads (QV 20 or higher)
 - Lower-quality but still polished consensus reads (QV 1 - QV 20)
 - Unpolished consensus reads (RQ=-1)
 - 0- or 1-pass subreads unaltered (RQ=-1)

The `reads.bam` file should **not** be used by itself as input for non-SMRT Link tools that expect \geq QV 20. The BAM format is a binary, compressed, record-oriented container format for raw or aligned sequence reads. The associated SAM format is a text representation of the same data. The BAM specifications are maintained by the SAM/BAM Format Specification Working Group. BAM files produced by **all** Sequel II systems are **fully compatible** with the BAM specification. For more information on the BAM file format specifications, click [here](#).

- `hifi.reads.bam.pbi`: Index file that allows for random access of HiFi reads in the BAM file.
- `sts.xml`: Contains summary statistics about the collection/cell and its post-processing.
- `zmw_metrics.json.gz`: Contains processing information used to generate RunQC plots.
- `5mc_report.json`, `primrose.log`: Contains information about 5mC CpG Detection analysis (using the `primrose` tool), if performed.
- `<Barcode Name>.consensusreadset.xml`: Contains reads associated with a specific barcode.
- `<Barcode Name>.bam`: Contains HiFi reads associated with a specific barcode, in .bam format.
- `<Barcode Name>.bam.pbi`: Index file that allows for random access of HiFi reads in the BAM file.
- `<Barcode Name>.fasta`: Contains reads associated with the specific barcode, in FASTA format.
- `lima.log`: Log file from the demultiplexing analysis, if performed. Informative for debugging and performance tracking by PacBio.
- `lima.counts.txt`: Contains the counts of each observed barcode pair. Only passing ZMWs are counted.

- `lima.guess.json`, `lima.guess.txt`: Describes the barcode subsetting process activated using the `--peek` and `--guess` options. These files contain the same information, and are used internally by PacBio Technical Support.
- `lima.reports.txt`: A tab-separated file describing each ZMW, unfiltered. This is useful information for investigating the demultiplexing process and the underlying data. A single row contains **all** reads from a single ZMW.
- `lima.summary.txt`: Lists how many ZMWs were filtered, how many ZMWs are the same or different, and how many reads were filtered.
- `unbarcoded.consensusreadset.xml`, `unbarcoded.hifi_reads.bam`, `unbarcoded.hifi_reads.bam.pbi`: Contains information on HiFi reads **not** associated with any barcode.

Note: The Sequel Ie system runs CCS on-instrument by default and the `subreads.bam`, `subreads.bam.pbi`, `scraps.bam` and `scraps.bam.pbi` files are no longer generated and are **not** available. Even though the `subreads.bam` and `subreads.bam.pbi` files are **not** accessible by default, there is a mechanism available to enable their output. For detailed instructions on how to enable the output of these files, contact your Field Applications Support team members.

HiFi reads generation

A standard Run Design performs on-instrument CCS analysis **without** including low-quality reads and generates a `hifi_reads.bam` file and transfers it to the network server. If low-quality reads are included in the Run Design, the Sequel Ie system will produce a `reads.bam` file, which contains HiFi reads and non-HiFi reads, and should **not** be used unfiltered as input for tools that expect \geq QV 20. SMRT Link **automatically** launches an **Export Reads** job on the `reads.bam` to filter out the HiFi reads, and generates the following HiFi data files by default:

- `<Movie_Name>.hifi_reads.fastq.gz` - Gzipped FASTQ file containing HiFi reads.
- `<Movie_Name>.hifi_reads.fasta.gz` - Gzipped FASTA file containing HiFi reads.
- `<Movie_Name>.hifi_reads.bam` - BAM file containing HiFi reads.

If **not** using SMRT Link for subsequent analysis, use these three files as input with any third-party analysis tools.

Finding the `hifi_reads` files generated using On-Instrument CCS

1. In Run QC, click the desired run, then click the sample name to view the CCS Data Set.
2. Click **Analyses** in the left-side panel.
3. Click the **Export Reads** analysis.

Dataset Overview

Adapter Report

Raw Data Report

CCS Analysis Report

Completed Analyses

Name	State	Id	Date Created	Created By	Analysis Application
Auto Analyses of 64009 0211 SAT OICCS	SUCCESSFUL	27671	2021-02-11, 01:45:49 PM	aswei	
Export Reads of 2kb Lambda OICCS	SUCCESSFUL	27672	2021-02-11, 01:45:49 PM	aswei	Export Reads

4. To locate the directory containing the three `hifi_reads` files, append `/outputs` to the path shown.

▼ Analysis Overview

Status

Display All

▶ Data

Analysis

Export Reads of 2kb Lambda OICCS

Analysis ID

27672

From Multi-Job

27671

Status

SUCCESSFUL: 5 tasks finished

Created By

aswei

Date Created

2021-02-11, 01:45:49 PM

Date Updated

2021-02-11, 08:34:01 PM

Application

Export Reads

SMRT Link Version

10.1.0.115913

Inputs

Data Type	Name	Import Complete
ConsensusReadSet	HIFI Reads: 2kb Lambda OICCS-Cell1 [...]	Yes

Path

/jpbi/dept/secondary/siv/smartlink/smartlink-siv-alpha/smartlink_5.1.0.SNAPSHOT13617/userdata/jobs_root/0000/0000027/0000027672

Sequel II system output files

Following is a sample of the file and directory structure output by the Sequel II system:

```
<your_specified_output_directory>/r64008_20160116_003347/1_A01
|-- m64008_160116_003634.baz2bam_1.log
|-- m64008_160116_003634.scrap.bam
|-- m64008_160116_003634.scrap.bam.pbi
|-- m64008_160116_003634.subreads.bam
|-- m64008_160116_003634.subreads.bam.pbi
|-- m64008_160116_003634.subreadset.xml
|-- m64008_160116_003634.sts.xml
|-- m64008_160116_003634.transferdone
```

Files output by the Sequel II system include:

- `scrap.bam` and `scrap.bam.pbi`: These files contain sequence data outside of the high-quality region, rejected subreads, excised adapter and possible barcode sequences, as well as spike-in control sequences. (The basecaller marks regions of single molecule sequence activity as high-quality.) **Note**: This applies to files generated by Sequel Instrument Control Software (ICS) v3.1.0 or later.
- `subreads.bam`: The Sequel II system output **one** `subreads.bam` file per collection/cell, which contains unaligned base calls from high-quality regions. This file is transferred from the instrument to network storage, then is used as **input** for secondary analysis by PacBio's SMRT Analysis software. Data in a `subreads.bam` file is analysis-ready; all of the data present should be quality-filtered for analyses. Subreads that contain information such as double-adapter inserts or single-molecule artifacts are **not** used in secondary analysis, and are excluded from this file and placed in `scrap.bam`.

- `subreads.bam.pbi`: Provides backwards-compatibility with the APIs enabled for accessing the `cmp.h5` file.
- `subreadset.xml`: This file is needed to import data into SMRT Link.
- `sts.xml`: Contains summary statistics about the collection/cell and its post-processing.
- `transferdone`: Contains a list of files successfully transferred.

Frequently asked questions

What are the minimum files needed to analyze data on SMRT Link?

- `.bam` file
- `bam.pbi` file
- `subreadset.xml` or `consensusreadset.xml` file

What is the average size of the file bundle for a 30-hour movie of HiFi reads?

Approximately 50 Gb.

What is the difference between a regular `.bam` file and an `aligned.bam` file?

The `subreads.bam` file contains all the subreads sequences, while the `aligned.bam` file additionally contains the genomic coordinates of the reads mapped to a reference sequence.

The `subreads.bam` file is created by the Sequel II systems, while the `aligned.bam` file is created by SMRT Link after running mapping analysis applications.

Secondary analysis output files

This is data produced by secondary analysis, which is performed on the primary analysis data generated by the instrument.

- All files for a specific job reside in **one** directory named according to the job ID number.
- Every job result has the following file structure. **Example:**

```
$SMRT_ROOT/userdata/jobs_root/0000/00000000/00000000002/
├── cromwell-job -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│   └── pb_demux_subreads_auto/24e691c8-8d0d-4670-9db3-c7cb1126e8f8
│       ├── entry-points
│       │   └── ae6f1c2c-b4a2-41cc-8e44-98b494f12a57.subreadset.xml
│       ├── logs
│       │   ├── pb_simple_mapping
│       │   │   └── 24e691c8-8d0d-4670-9db3-c7cb1126e8f8
│       │   │       ├── call-mapping
│       │   │       │   └── execution
│       │   │       │       ├── stderr
│       │   │       │       └── stdout
│       │   └── workflow.24e691c8-8d0d-4670-9db3-c7cb1126e8f8.log
│       ├── outputs
│       │   ├── mapping.report.json -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│       │   │   └── pb_simple_mapping/24e691c8-8d0d-4670-9db3-c7cb1126e8f8/call-mapping/execution/
│       │   │       mapping.report.json
│       │   └── mapped.bam -> $SMRT_ROOT/userdata/jobs_root/cromwell-executions/
│       │       └── pb_simple_mapping/24e691c8-8d0d-4670-9db3-c7cb1126e8f8/call-mapping/execution/
│       │           mapped.bam
│       ├── pbscala-job.stderr
│       ├── pbscala-job.stdout
│       └── workflow
│           ├── analysis-options.json
│           ├── datastore.json
│           ├── engine-options.json
│           ├── inputs.json
│           ├── metadata.json
│           ├── metadata-summary.json
│           ├── task-timings.metadata.json
│           └── timing-diagram.html
```

- **logs/:** Contains log files for the job.
 - workflow.<UUID>.log: Global log of each significant step in the job and snippets from a task's `stderr` output if the job failed.
 - The same directory contains `stdout` and `stderr` for individual tasks.
- **cromwell-job/:** Symbolic link to the actual Cromwell execution directory, which resides in another part of the `jobs-root` directory. Contains subdirectories for each workflow task, along with executable scripts, output files, and `stderr/stdout` for the task.
 - call-tool_name/execution/: Example of an individual task directory (This is replaced with <task_id> below.)
 - <task_id>/stdout: General task `stdout` log collection.
 - <task_id>/stderr: General task `stderr` log collection.

- `<task_id>/script`: The SMRT Tools command for the given analysis task.
- `<task_id>/script.submit`: The JMS submission script wrapping `run.sh`.
- `<task_id>stdout.submit`: The `stdout` collection for the `script.submit` script.
- `<task_id>/stderr.submit`: The `stderr` collection for the `script.submit` script.
- `workflow/`: Contains JSON files for job settings and workflow diagrams.
 - `datastore.json`: JSON file representing all output files imported by SMRT Link.
- `outputs/`: A directory containing symbolic links to all datastore files, which reside in the Cromwell execution directory. This is provided as a convenience and is **not** intended as a stable API; note that external resources from dataset XML and report JSON file are **not** included here. Demultiplexing outputs are nested in additional subdirectories.
- `pbscala-job.stderr`: Log collection of `stderr` output from the SMRT Link job manager.
- `pbscala-job.stdout`: Log collection of `stdout` output from the SMRT Link job manager. (**Note**: This is the file displayed as **Data > SMRT Link Log** on the analysis results page.)

A SMRT Link job generates several types of output files. You can use these data files as input for further processing, pass on to collaborators, or upload to public genome sites. Depending on the analysis application being used, the `output` directory contain files in the following formats:

- **BAM**: Binary version of the Sequence Alignment Map (SAM) format. (See [here](#) for details.)
- **BAI**: The `samtools` index file for a file generated in the BAM format.
- **BED**: Format that defines the data lines displayed in an annotation track. (See [here](#) for details.)
- **CSV**: Comma-Separated Values file. Can be viewed using Microsoft Excel or a text editor.
- **FASTA/FASTQ**: Sequence files that contains either nucleic acid sequence (such as DNA) or protein sequence information. FASTA/Q files store multiple sequences in a single file. FASTQ files also include per-base quality scores. (See [here](#) or [here](#) for details.)
- **GFF**: General Feature Format, used for describing genes and other features associated with DNA, RNA and protein sequences. (See [here](#) for details.)
- **PBI**: PacBio index file. (This is a PacBio-specific file type.)
- **VCF**: Variant Call Format, for use with the molecular visualization and analysis program VMD. (See [here](#) for details.)

To download data files created by SMRT Link:

1. On the home page, select **SMRT Analysis**. You see a list of **all** jobs.
2. Click the job link of interest.
3. Click **Data > File Downloads**, then click the appropriate file. The file is downloaded according to your browser settings.
 - **(Optional)** Click the small icon to the right of the file name to copy the file's path to the Clipboard.

Configuration and user management

LDAP

SMRT Link supports the use of LDAP for user login and authentication. **Without** LDAP integration with SMRT Link, only **one** user (with the login `admin/admin`) is enabled. You can add new users **after** SMRT Link is integrated and configured to work with LDAP; you can also add new users using WSO2 API Manager or Keycloak **without** LDAP integration.

- For details on integrating LDAP and SMRT Link, see the document **SMRT Link software installation guide (v11.1)**.

SSL

SMRT Link requires the use of Secure Sockets Layer (SSL) to enable access via HTTP over SSL (HTTPS), so that SMRT Link logins and data are encrypted during transport to and from SMRT Link. SMRT Link includes an Identity Server (WSO2 API Manager or Keycloak), which can be configured to integrate with your LDAP/AD servers and enable user authentication using your organizations' user name and password. To ensure a secure connection between the SMRT Link server and your browser, the SSL certificate can be installed **after** completing SMRT Link installation.

It is important to note that PacBio will **not** provide a signed SSL certificate, however – once your site has obtained one – PacBio tools can be used to install it and configure SMRT Link to use it. You will need a certificate issued by a Certificate Authority (CA, sometimes referred to as a **certification authority**). PacBio has tested SMRT Link with certificates from the following certificate vendors: VeriSign, Thawte and digicert.

Note: PacBio recommends that you consult your IT administrator about obtaining an SSL certificate.

Alternatively, you can use your site's self-signed certificate.

SMRT Link ships with a PacBio self-signed SSL certificate. If used, **each** user will need to accept the browser warnings related to access in an insecure environment. Otherwise, your IT administrator can configure desktops to **always** trust the provided self-signed certificate. Note that SMRT Link is installed within your organization's secure network, behind your organization's firewall.

- For details on updating SMRT Link to use an SSL certificate, see the document **SMRT Link software installation guide (v11.1)**.

The following procedures are available **only** for SMRT Link users whose role is **Admin**.

Adding and deleting SMRT Link users

1. Choose **Gear > Configure**, then click **User Management**.
2. There are two ways to find users:
 - To display **all** SMRT Link users: Click **Display all Enabled Users**.
 - To find a specific user: Enter a user name, or partial name, and click **Search By Name**.
3. Click the desired user. If the user status is **Enabled**, the user has access to SMRT Link; **Disabled** means the user **cannot** access SMRT Link.
 - To **add** a SMRT Link user: Click the **Enabled** button, then assign a role. (See below for details.)
 - To **disable** a SMRT Link user: Click the **Disabled** button.
4. Click **Save**.

Assigning user roles

SMRT Link supports three user roles: **Admin**, **Lab Tech**, and **Bioinformatician**. Roles define which SMRT Link modules a user can access. The following table lists the privileges associated with the three user roles:

Tasks/privileges	Admin	Lab Tech	Bioinformatician
Add/delete SMRT Link users	Y	N	N
Assign roles to SMRT Link users	Y	N	N
Update SMRT Link software	Y	N	N
Access Sample Setup module	Y	Y	N
Access Run Design module	Y	Y	N
Access Run QC module	Y	Y	Y
Access Data Management module	Y	Y	Y
Access SMRT Analysis module	Y	Y	Y

1. Choose **Gear > Configure**, then click **User Management**.
2. There are two ways to find users:
 - To display **all** SMRT Link users: Click **Display all Enabled Users**.
 - To find a specific user: Enter a user name, or partial name, and click **Search By Name**.
3. Click the desired user.
4. Click the **Role** field and select one of the three roles. (A **blank** role means that this user **cannot** access SMRT Link.)

- **Note:** There can be **multiple** users with the Admin role; but there **must** always be at least **one** Admin user.
5. Click **Save**.

Hardware/software requirements

Client hardware requirements

- SMRT Link requires a minimum screen resolution of 1600 by 900 pixels.

Client software requirements

- SMRT Link **requires** the Google® Chrome web browser, version 90 or later.

Note: SMRT Link **server** hardware and software requirement are listed in the document **SMRT Link software installation guide (v11.1)**.

Appendix A - PacBio terminology

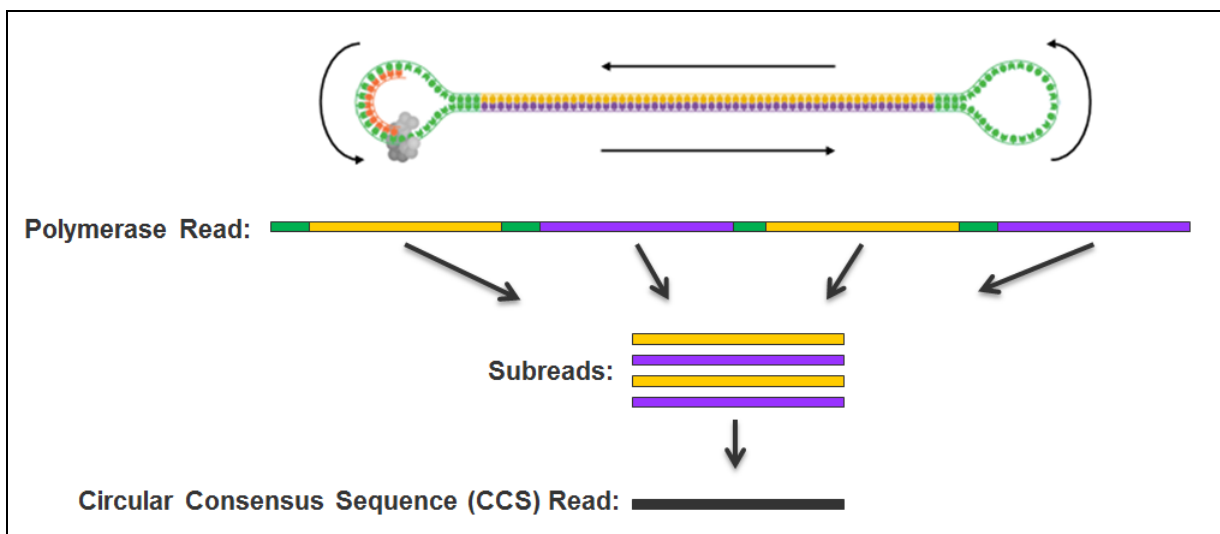
General terminology

- **SMRT[®] Cell**: Consumable substrates comprising arrays of zero-mode waveguide nanostructures. SMRT Cells are used in conjunction with the DNA sequencing kit for on-instrument DNA sequencing.
- **SMRTbell[®] template**: A double-stranded DNA template capped by hairpin adapters (i.e., SMRTbell adapters) at both ends. A SMRTbell template is topologically circular and structurally linear, and is the library format created by the DNA template prep kit.
- **collection**: The set of data collected during real-time observation of the SMRT Cell; including spectral information and temporal information used to determine a read.
- **Zero-mode waveguide (ZMW)**: A nanophotonic device for confining light to a small observation volume. This can be, for example, a small hole in a conductive layer whose diameter is too small to permit the propagation of light in the wavelength range used for detection. Physically part of a SMRT Cell.
- **Run Design**: Specifies
 - The samples, reagents, and SMRT Cells to include in the sequencing run.
 - The run parameters such as movie time and loading to use for the sample.
- **adaptive loading**: Uses active monitoring of the ZMW loading process to predict a favorable loading end point.
- **unique molecular yield**: The sum total length of unique single molecules that were sequenced. It is calculated as the sum of per-ZMW median subread lengths.

Read terminology

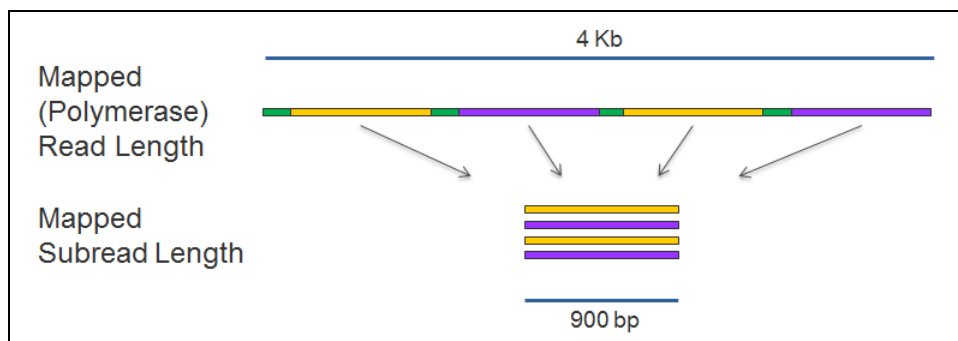
- **polymerase read**: A sequence of nucleotides incorporated by the DNA polymerase while reading a template, such as a circular SMRTbell template. They can include sequences from adapters and from one or multiple passes around a circular template, which includes the insert of interest. Polymerase reads are most useful for quality control of the instrument run. Polymerase read metrics primarily reflect movie length and other run parameters rather than insert size distribution. Polymerase reads are trimmed to include only the high-quality region. **Note**: Sample quality is a major factor in polymerase read metrics.
- **subreads**: Each polymerase read is partitioned to form one or more subreads, which contain sequence from a single pass of a polymerase on a single strand of an insert within a SMRTbell template and no adapter sequences. The subreads contain the full set of quality values and kinetic measurements. Subreads are useful for applications such as *de novo* assembly, base modification analysis, and so on.
- **longest subread length**: The mean of the maximum subread length per ZMW.

- **insert length:** The length of the double-stranded nucleic acid fragment in a SMRTbell template, excluding the hairpin adapters.
- **circular consensus (CCS) reads:** The consensus sequence resulting from alignment between subreads taken from a single ZMW. Generating CCS reads does **not** include or require alignment against a reference sequence but **does** require at least two full-pass subreads from the insert. CCS reads are generated with CCS analysis. CCS reads with quality value equal to or greater than 20 are called **HiFi reads**.
- **HiFi reads:** Reads generated with CCS analysis whose quality value is equal to or greater than 20.



Read length terminology

- **mapped polymerase read length:** Approximates the sequence produced by a polymerase in a ZMW. The total number of bases along a read from the first adapter of aligned subread to the last adapter or aligned subread.
- **mapped subread length:** The length of the subread alignment to a target reference sequence. This does **not** include the adapter sequence.



Secondary analysis terminology

- **secondary analysis:** Follows primary analysis and uses basecalled data. It is application-specific, and may include:

- Filtering/selection of data that meets a desired criteria, such as quality, read length, and so on.
- Comparison of reads to a reference or between each other for mapping and variant calling, consensus sequence determination, alignment and assembly (*de novo* or reference-based), variant identification, and so on.
- Quality evaluations for a sequencing run, consensus sequence, assembly, and so on.
- PacBio's SMRT Analysis contains a variety of secondary analysis applications including RNA and Epigenomics analysis tools.
- **secondary analysis application:** A secondary analysis workflow that may include multiple analysis steps. Examples include *de novo* assembly, RNA and epigenomics analysis.
- **consensus:** Generation of a consensus sequence from multiple-sequence alignment.
- **filtering:** Removes reads that do not meet the Read Length criteria set by the user.
- **mapping:** Local alignment of a read or subread to a reference sequence.
- **Auto Analysis:** Allows a specific analysis to be automatically run after a sequencing run has finished and the data is transferred to the SMRT Link server. The analysis can include demultiplexed outputs.
 - Auto Analysis works with **all** Sequel II systems and Sequel IIe systems.
- **Pre Analysis:** The process of CCS analysis and/or demultiplexing of Sequel basecalled data. Pre Analysis occurs **before** Auto Analysis.
 - Pre Analysis works with **all** Sequel II systems and Sequel IIe systems.

Accuracy terminology

- **circular consensus accuracy:** Accuracy based on consensus sequence from multiple sequencing passes around a single circular template molecule.
- **consensus accuracy:** Accuracy based on aligning multiple sequencing reads or subreads together.
- **polymerase read quality:** A trained prediction of a read's mapped accuracy based on its pulse and base file characteristics (peak signal-to-noise ratio, inter-pulse distance, and so on).

Appendix B - Data search

Use this function to search for jobs, Data Sets, barcode files, or reference files.

To search the entire table

1. Enter a text query into the Search box. This searches **every field** in the table, and displays **all** table rows containing the search characters.

To search for a value within a column

1. Click the small filter icon at the right of the column name.
2. Enter a value; **all** table rows meeting the search criteria display. (To select a **different** search operator, click the droplist and select another search operator. Different search operators are available, based on the column's data type.)

- For the **Analysis State** column only, click one or more of the job states of interest: **Select All, Created, Running, Submitted, Terminated, Successful, Failed, or Aborted.**
- For **Date fields** only, click the small calendar and select a date.

Numeric field operators

- Equals, Not equal
- Greater than, Greater than or equals

- Less than, Less than or equals
- In range

Text field operators

- Contains, Not contains
- Equals, Not equal
- Starts with, Ends with

Date field operators

- Equals, Not equal
- Greater than, Less than
- In range

Appendix C - BED file format for Target Regions report

With the HiFi Mapping application, an optional **Target Regions** report can be generated that displays the number (and percentage) of reads and subreads that hit specified target regions.

The BED file required to generate the Target Regions report includes the following fields; with one entry per line:

1. **chrom**: The name of the chromosome (such as `chr3`, `chrY`, `chr2_random`) or scaffold (such as `scaffold10671`).
2. **chromStart**: The starting position of the feature in the chromosome or scaffold. The first base in a chromosome is numbered 0.
3. **chromEnd**: The ending position of the feature in the chromosome or scaffold. The **chromEnd** base is **not** included in the display of the feature, however, the number in position format is represented. For example, the first 100 bases of chromosome 1 are defined as `chrom=1, chromStart=0, chromEnd=100`, and span the bases numbered 0-99 (**not** 0-100), but will represent the position notation `chr1:1-100`.
4. **(Optional) Region Name**.

Example: `lambda_NEB3011 15000 25000 Region2`

- Fields can be space- or tab-delimited.
- See [here](#) for details of the BED format.
- For details on the BED format's counting system, see [here](#) and [here](#).

Appendix D - Additional information included in the CCS Data Set Export report

When you export a Data Set and select **Export PDF Reports**, a report is produced which includes additional fields, listed below.

- See [“Exporting sequence, reference and barcode data” on page 42](#) for details on exporting Data Sets.
- The other fields and plots in this report are described in the appropriate Reports sections of [“PacBio® secondary analysis applications” on page 54](#) and [“PacBio® data utilities” on page 99](#).
- **ZMWs input:** The total number of ZMWs used as input in the Data Set.
- **ZMWs pass filters:** The number of ZMWs that passed **all** the filters.
- **ZMWs fail filters:** The number of ZMWs that failed **any** of the filters.
- **ZMWs shortcut filters:** The number of low-pass ZMWs skipped using the `--all` filter.
- **ZMWs with tandem repeats:** The number of ZMWs that did not generate CCS reads due to repeats larger than `--min-tandem-repeat-length`.
- **Below SNR threshold:** The number of ZMWs that did not generate CCS reads due to SNR below `--min-snr`.
- **Median length filter:** The number of ZMWs that did not generate CCS reads due to subreads that are <50% or >200% of the median subread length.
- **Lacking full passes:** The number of ZMWs that did not generate CCS reads due to having fewer than `--min-passes` full-length subreads.
- **Heteroduplex insertions:** The number of ZMWs that did not generate CCS reads due to single-strand artifacts.
- **Coverage drops:** The number of ZMWs that did not generate CCS reads due to coverage drops that would lead to unreliable polishing results.
- **Insufficient draft cov:** The number of ZMWs that did not generate CCS reads due to not having enough subreads aligned to the draft sequence end-to-end.
- **Draft too different:** The number of ZMWs that did not generate CCS reads due to having fewer than `--min-passes` full-length reads aligned to the draft sequence.
- **Draft generation error:** The number of ZMWs that did not generate CCS reads due to subreads that don't agree enough to generate a draft sequence.
- **Draft above --max-length:** The number of ZMWs that did not generate CCS reads due to a draft sequence longer than `--max-length`.
- **Draft below --min-length:** The number of ZMWs that did not generate CCS reads due to a draft sequence shorter than `--min-length`.
- **Reads failed polishing:** The number of ZMWs that did not generate CCS reads due to too many subreads dropped while polishing.

- **Empty coverage windows:** The number of ZMWs that did not generate CCS reads because at least one window had no coverage.
- **CCS did not converge:** The number of ZMWs that did not generate CCS reads because the draft sequence had too many errors that could not be polished in time.
- **CCS below minimum RQ:** The number of ZMWs that did not generate CCS reads because the predicted accuracy is below `--min-rq`.
- **Unknown error:** The number of ZMWs that did not generate CCS reads due to rare implementation errors.